INTEGRATING AUTOMATED MULTI-DISCIPLINARY OPTIMIZATION IN PRELIMINARY DESIGN OF NON-TRADITIONAL AIRCRAFT

THESIS

Mehmet Fidanci
First Lieutenant, TUAF

Jeffrey R. Miller
Captain, USAF

Douglas J. Strauss
Captain, USAF

AFIT/GSE/ENY/00M-01

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GSE/ENY/00M-01

INTEGRATING AUTOMATED MULTI-DISCIPLINARY OPTIMIZATION IN PRELIMINARY DESIGN OF NON-TRADITIONAL AIRCRAFT

THESIS

Presented to the Faculty

Department of Aeronautics

Graduate School of Engineering and Management

of the Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Systems Engineering

Mehmet Fidanci
First Lieutenant, TUAF

Jeffrey R. Miller
Captain, USAF

Douglas J. Strauss
Captain, USAF

March 2000

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AFIT/GSE/ENY/00M-1

INTEGRATING AUTOMATED MULTI-DISCIPLINARY OPTIMIZATION IN PRELIMINARY DESIGN OF NON-TRADITIONAL AIRCRAFT

Mehmet Fidanci, B.S.
1st Lt, TUAF

Jeffrey R. Miller, B.S.
Captain, USAF

Douglas J. Strauss, B.S.
Captain, USAF

Approved:

Ernest P. Smith

Ernest P. Smith (Chairman) — 7 March 00 date

David R. Jacques

David R. Jacques (Member) — 7 MAR 00 date

Curtis H. Spenny

Curtis H. Spenny (Member) — 7 March '00 date

ACKNOWLEDGMENTS

We would like to express our appreciation to Dr Maxwell Blair, Ms. Victoria Tischler, Dr. Vipperla Venkayya, and Dr. Jeffrey Zweber of AFRL/VA for teaching us and helping us so much, and for putting up with us. We also would like to express our sincere thanks to Alok Mathur, John Marino, and others at TSI who provided AML training, the numerous versions of AML, and an inordinate amount of customer support for the Team. We would also like to thank our friends and families who helped us and greatly inconvenienced themselves for our sakes through this time at AFIT.

Mehmet Fidanci
Jeff Miller
Doug Strauss

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

AFIT/GSE/ENY/00M-01

ABSTRACT

Current methods of aircraft conceptual design lack the ability to quickly generate detailed analysis, particularly of nontraditional designs such as blended wing body craft. This study developed a method to resolve this problem by creating a flexible, parametrically driven conceptual model in an object-oriented, adaptive modeling environment from which analysis and optimization may rapidly be performed. These object-oriented techniques are incorporated into a traditional conceptual design process. All objects inherit dependency-tracking and demand-driven calculations.

Design Analysis was performed within the modeling language and utilized interfaces to other software packages. A detailed mesh, suitable for input into finite element analysis programs, was developed from the less detailed, geometric mesh created by the modeling program. The output from finite element analysis forms the basis for rapid changes in subsequent iterations of the design process.

The demonstration focuses on a single parametric design model which transforms a conventional transport design into a blended wing body design. This single design is controlled by a limited set of geometric variables and produces optimal structural weight estimations while the designer addresses volumetric and cost requirements.

INTEGRATING AUTOMATED MULTI-DISCIPLINARY OPTIMIZATION IN PRELIMINARY DESIGN OF NON-TRADITIONAL AIRCRAFT

1. INTRODUCTION

1.1 Background

As the 20^{th} century comes to a close, global competition among aircraft manufacturers has increased. Aircraft designers and manufacturers are constantly improving aircraft in many areas including weight, range, cost, noise, etc. In order to design more efficient aircraft, designers must iteratively engage the aircraft design process. Any improvements which designers contribute in each cycle of the process are fed back into subsequent cycles in order to achieve greater efficiency.

Transport aircraft have been incrementally improved based on the same underlying design for over fifty years. The traditional cigar shaped fuselage with wings attached to the sides has remained essentially unchanged. Recently the Blended Wing Body (BWB) aircraft design has again re-surfaced as a new idea which offers many potential savings in efficiency and lift.

However, to take advantage of the BWB design many obstacles must be overcome. The BWB has a non-circular fuselage with a pressurized interior cabin that is very difficult to analyze structurally. The aircraft design community is unsure of how exactly to use finite element analysis (FEA) to model a BWB design. It is also very unclear how the manufacturing processes for a BWB design would occur.

Current aircraft designs are very complex and integrated. Many steps must take place in their design involving multiple disciplines. Communication among them is extremely difficult. In addition, when communication among disciplines is not pursued, possible synergies go unrealized. An example is the blended wing body concept where the fuselage provides active lift and the wing provides the cargo area traditionally supplied by the fuselage. A BWB aircraft is aerodynamically efficient. The aircraft design effort succeeds with integrated aerodynamic and cargo-lift groups; however, if an aircraft design effort was split into separate wing and cargo-carrying groups, neither group would have had an incentive to help the other.

A solution to the communications challenge is for the entire aircraft design team to work on a single computerized design system. A common database ensures that all members are using the correct information for their work. Computations can be more in-depth, since considerably more information is available when the design of the entire plane is available.

In order for more in-depth analysis to be performed, however, the initial aircraft design must be modeled to a high degree of definition, typically using a finite element method (FEM) software package. The model used in FEM typically is difficult to produce and extremely difficult to change. Model development is a multi-step process of which little is computerized. FEM begins with the designer defining the geometry of interest and determining how to subdivide the geometry (quadrilateral elements, size of elements, etc.) The designer then determines the locations of the nodes, a set of points with specific, physical meanings, for example, intersections between elements. The designer inputs the locations of the nodes and then defines the connectivity of the

elements formed by them. The software package assists the designer in creating the FE model by graphically displaying the location of the user-defined nodes (Strutzenberg, 1999). The FE model is then sent to a finite element analysis (FEA) or optimization program. Changes to the model require redefining node points and the connectivity between them, which often requires a complete reworking of the model.

Because of the time-intensive nature of developing a model and the limited flexibility of an FEM, designers often invest large amounts of time performing design work before developing a FE model. The analysis and optimization power of programs which rely on FEM for input is lost to conceptual designers who cannot or are unwilling to invest the time in creating a model. This is a great hurdle to streamlining and improving the aircraft design process.

With rapid finite element modeling, the effort involved in developing a FE model is significantly reduced. This allows more information to be available earlier in the design cycle than would otherwise occur, and thereby reduces risk and uncertainty involved with conceptual design.

The Air Force Research Laboratory Air Vehicles Directorate (AFRL/VA) is the Air Force's office responsible for developing technology for aircraft. The ability to explore and analyze technologies in the context of conceptual designs of aircraft is important to the mission of the Directorate. Having credible models of complex systems aids the Directorate in promoting technology programs.

1.2 Problem Statement

The specific task of the Systems Engineering Design Team was to develop a method of conceptual aircraft design which would take advantage of an object-oriented modeling program interfaced with a meshing program to generate a finite element model from a conceptual aircraft model. The FE model would then be used to form an input file for a finite element analysis program which would return the optimized weight of the model, providing insights about the aircraft design and aircraft design process .

1.3 Problem Solution

Object-oriented computer software packages have appeared which greatly simplify the task of modeling an aircraft design. Using such a package produced by TechnoSoft, Incorporated (TSI) called the Adaptive Modeling Language (AML), it is possible to rapidly generate a parametric model of a candidate aircraft design. The parametric model incorporates many aspects of the aircraft design. Parametric modeling controls many aircraft variables in terms of a few primary dimensions. This subsequently simplifies experimental design changes in the candidate aircraft planform. Utilizing such a model, when a change in one variable is made, it automatically causes design changes and ripples through the entire design. This encourages rapid improvement, evaluation, feedback and re-evaluation of many subsequent aircraft designs. Trade studies may then be generated based on a much greater information base.

AFRL/VA's Multidisciplinary Technology Center (MDT) has documented their experience in using AML to generate parametrically driven design and scenario objects and integrating design scenarios with conceptual vehicle designs (Blair, 1998; Blair and

others, 1997). The MDT Center has sponsored recent AML object developments that interface directly to a meshing software package, MSC.PATRAN. MSC.PATRAN is used to generate a geometric mesh of the AML-created model. From MSC.PATRAN's output files, an appropriate input data file may be constructed for the FEA aeronautical evaluation/optimization software package, Automated Structural Optimization System (ASTROS). This effort seeks to construct a process that will quickly progress from concept to evaluation with far faster feedback and flexibility of design change. This model demonstrates the practicality of using an object-oriented modeling language to generate finite element analysis for conceptual designs.

1.4 Scope of Effort

This thesis effort focused on developing a method to use certain software tools in order to gain more information about a particular conceptual aircraft design than would be available using traditional methods of aircraft conceptual design. In pursuit of this, the team developed several highly flexible software objects and successfully demonstrated mesh generation using MSC.PATRAN from within AML for the first time on an AFRL machine.

This thesis effort was undertaken with the following goals: (1) to develop a simple, parametrically-driven, geometric model of an aircraft in AML using a minimum of station locations and a limited number of parameters, while allowing for the design to be morphed into nontraditional shapes, such as a BWB. (2) to integrate an AML object for automated generation of an FEA model of the above model. (3) to extract the geometric mesh information from the AML model. (4) to convert the geometric

connectivity files into a format acceptable to a finite element analysis (FEA) program. (5) to submit the revised FEA input file, to include appropriate loadings, into a FEA program and have analysis/optimization successfully performed on it. (6) to establish an appropriate measure of the model's performance and rate it. The above six steps would result in one complete iteration of a design process. (7) to take the information gathered in the first six steps to change the aircraft model to improve the measure of performance. (8) to regenerate the geometric mesh, recreate the finite element mesh, and rerun the analysis programs to see what the results of the changes undertaken in (7) were.

1.5 Contribution of Research

At the same time that resources devoted to aircraft research and development are declining, the number of nontraditional aircraft designs and nontraditional aircraft missions the Air Force is being asked to evaluate is growing. With a reduction of resources occurring at the same time as an increase in scope, a real need exists to create tools that will allow the Air Force to evaluate concepts more quickly and yet more fully. The convergence of multi-disciplinary optimization (MDO) and computer-aided conceptual design is an opportunity to create these tools. The Air Vehicles Directorate was instrumental in developing ASTROS and has played a significant role in the development of AML; it is a natural sponsor for the integration of their functions.

This project was the first effort geared towards using a geometrically defined mesh as an input for an FEA program. As such, the project was first and foremost a "proof of concept" demonstration that the tools involved could work together. Demonstrating that a geometric model created in AML could be used in gathering FEA

information is the first step towards simplifying the amount of computer modeling required for an FEA of a conceptual design, or the first step towards gathering additional information about a conceptual design from a purely geometric model.

1.6 Sequence of Presentation

Chapter 2 provides a literature review of some of the relevant areas of conceptual aircraft design, multidisciplinary optimization, systems engineering theory, and computer tools used for conceptual design. Chapter 3 explains the methodology used by the Team to conduct the research for this effort. Chapter 4 details the results of the Team's research. Chapter 5 presents the conclusions of the research, with explanations of the limitations and areas for continued research. The appendices give more detailed information on the computer programs used, notes on the implementation of the software packages used, and the software codes written by the Team.

2. LITERATURE REVIEW

2.1 Current Method of Conceptual Design

2.1.1 Introduction. Aircraft design is a compromise of many requirements and technologies. These different design groups must work as a team to complete the best aircraft design. It is clear that the aircraft design process requires integration of many engineering disciplines such as aerodynamics, structures, flight controls, weights, stability, propulsion and other technical specialties.

Aircraft design is applied recursively on each process, as shown below in Figure 2-1. The design process is a complete development effort, beginning with general requirements and ending with a compliant product or process. Requirements are characteristics that identify the accomplishment levels needed to achieve specific objectives for certain conditions. Requirements are set by the former sizing and design trade studies. Requirements must be well understood otherwise the design team may be misdirected. If the requirements are inappropriate, the design will be ineffective in meeting its true goals. In order to meet requirements, design concepts are developed. These designs are analyzed in order to provide risk assessment, measure progress, evaluate design capabilities, and formulate and evaluate alternative courses of action. The most critical element in the design process is the design team leader who controls the system development effort with the goal of achieving an optimum balance of competing goals.

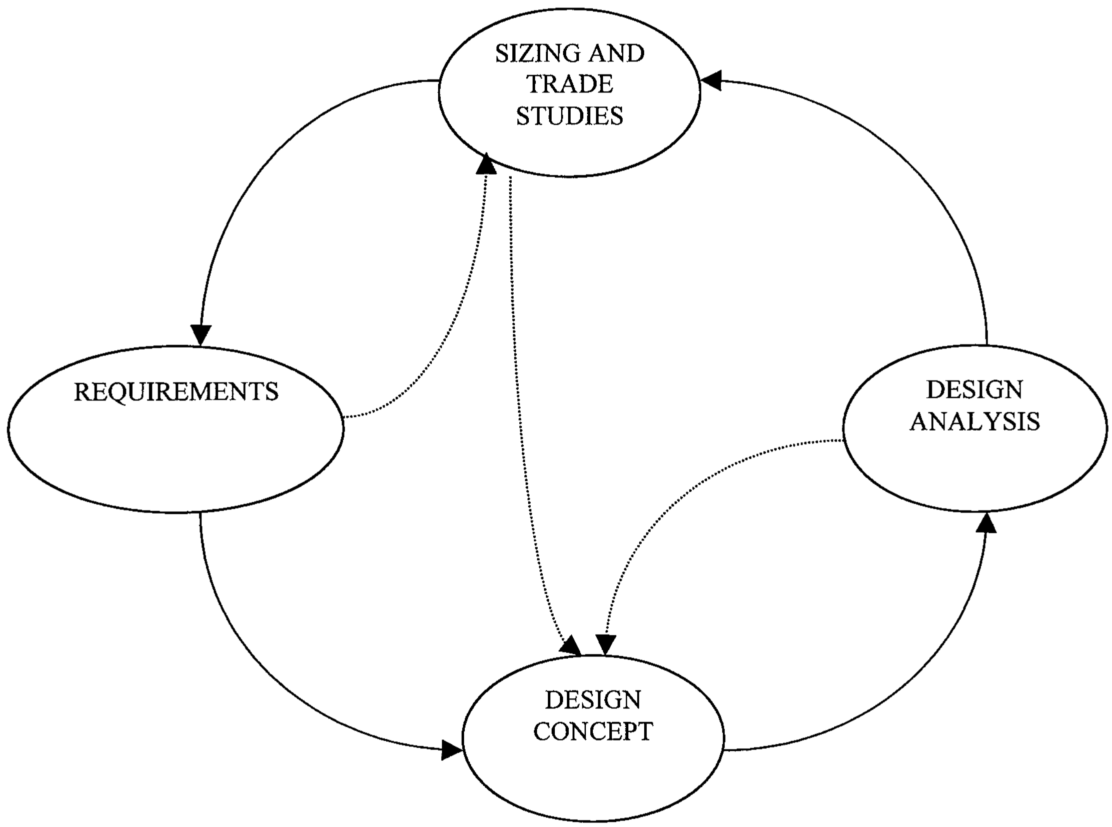


Figure 2-1: The Design Wheel.(Raymer, 1989)

2.1.2 Aircraft Design Process. The aircraft design process is usually divided into three phases or levels of design. These three phases are the conceptual design phase, preliminary design phase, and detail design phase.

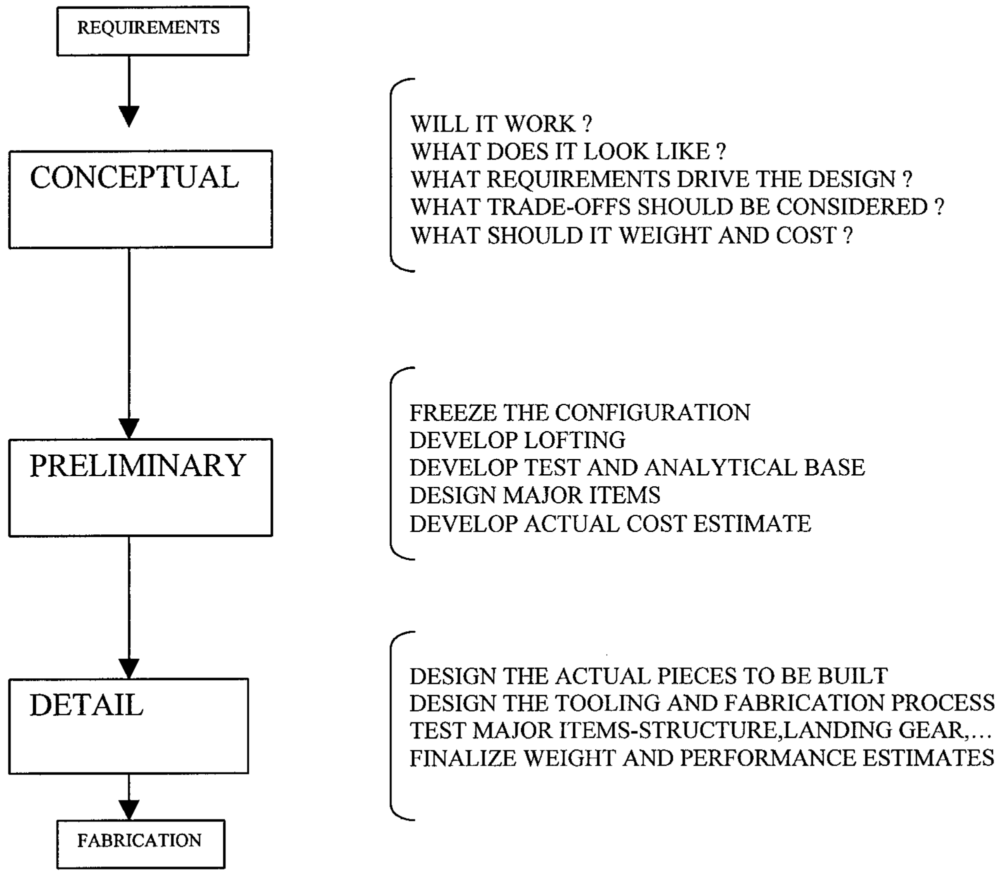


Figure 2-2: Three Phases Of Aircraft Design. (Raymer, 1989)

2.1.2.1 Conceptual Design Phase. Conceptual design phase begins with the specific set of mission requirements such as the aircraft total takeoff weight and payload, flight level and cruise speed, takeoff and landing distances and range requirements. The general configuration arrangement and size of the aircraft design is determined in this phase. Actually the design team is seeking what the aircraft design should look like and trying to answer the major 'what' questions shown in Figure 2.2.

Trade studies must be conducted on the best wing loading, wing sweep, aspect ratio, load-to-drag ratio, thickness ratio, fuselage shape and general wing-tail geometric

configurations using estimates of weights and aerodynamics. The design performance capabilities are calculated and compared to requirements. Some optimization techniques are used to perform the design mission while meeting all performance requirements.

According to Raymer, the conceptual design phase is a very fluid process. Each time the final design is analyzed, the latest design must be reestablished to reflect the new weight fractions such as total takeoff gross weight, fuel weight, wing size and engine size of the aircraft (Raymer, 1989). In addition to these factors, Nicolai mentions cost and manufacturing. A first look at cost and manufacturing must be made during the conceptual design phase. He also focuses on the feasibility of the design in order to achieve a given mission (Nicolai, 1975).

2.1.2.2 Preliminary Design Phase. Preliminary design is the most important phase in the aircraft design process. Studies have been performed investigating various design options. Because most of the work performed is only on paper, the cost is very minimal at this point. By looking at the preliminary design phase results, a company will decide to propose a full-scale development or not. The fabrication of the small parts may begin according to the results of the preliminary design phase. The preliminary design phase is more detailed than the conceptual design phase because the major changes in aircraft design are finished. All the different design groups in the design team will now analyze their portion of the aircraft.

Nicolai defines this phase as a fine-tuning of the conceptual design phase. This fine-tuning of the general configuration must be accomplished with a wind tunnel model. According to the results of aeroelastic, performance, fatigue and flutter analyses, some structural components might be built and tested. Weight and cost estimates must be

refined during the preliminary design phase. Dynamic stability and control influences on the control system also must be determined (Nicolai, 1975).

The key activity during the preliminary design phase is the mathematical modeling of the outside skin of the aircraft, which is called "lofting" by Raymer. This term involves sufficient accuracy to insure proper fit between its different parts, even if they are designed by different manufacurers (Raymer, 1989).

2.1.2.3 Detail Design Phase. Detail design phase is the final phase of the aircraft design process. This phase usually begins with the design of the actual pieces to be built and ends with the fabrication of the whole aircraft. Detailed structural design must be completed during this phase.

For example, the wing box object will be designed and analyzed as a whole during conceptual and preliminary design phases. During detail design phase, that whole wing box will be broken down into individual parts such as joints, fittings, attachments ribs, spars and skins. All these individual parts must be separately designed and analyzed (Raymer, 1989). It is clear that production design is the main part of the detail design phase. Specialists in the design groups determine how the airplane will be fabricated from smallest parts to the final assembly. Another important activity in the detail design phase is testing the major items. Flight simulators are used in this design phase to test the actual structure of the aircraft. Finally detail design phase ends with the fabrication of the whole aircraft.

2.1.3 Current Aircraft Conceptual Design Using PIANO Software. Current commercial aircraft conceptual design is embodied by the creation of the Fokker F-70, which was conceptually designed using the aircraft design software, Project Interactive

Analysis and Optimization (PIANO). PIANO is a professional software for the competitive analysis of both existing and projected commercial aircraft. It acts as the software that automates the conceptual design process found in Raymer's textbook and many others (Raymer, 1989). PIANO takes from 50 to 100 input parameters and produces a feasible aircraft design. PIANO is based on current experience with conventional circular fuselage aircraft. No finite element analysis is performed by PIANO because it is based on the vast experience with current conventional aircraft design. PIANO is not directly applicable to new aircraft designs such as the BWB because it is based on current, circular fuselage aircraft data and designs.

PIANO may be used in competitor evaluation, project sizing, performance estimation, and preliminary design studies. It can generate accurate, industrial-quality evaluations on a desktop or laptop Macintosh. The PIANO software is applicable to subsonic commercial designs ranging in size from small business aircraft (e.g. the Swearingen SJ30) to the largest airliners currently in service or projected for the next decade (e.g. the Airbus A3XX). PIANO can be used to generate new concept designs and to select promising candidates through point designs, parametric sensitivity studies, and multi-variate optimization (Simos, 1998). It is produced by Lyssys, a UK-based Aerospace Consultancy Company formed by Dr. Dimitri Simos. The origins of the PIANO followed post-doctoral research conducted in the mid-1980's at Loughborough University with the support of the Science Research Council and Short Brothers. PIANO has since evolved into an industrial tool used by many companies such as the Rolls-Royce plc (Derby) Aircraft Projects team (Simos, 1998).

The PIANO model produced complete sample outputs of the aircraft conceptual design process for the medium commercial transport, Fokker 70. The results are found in Appendix D and are known to match the manufacturer's claims quite well in areas where data are available. This is an independent analysis and does not necessarily reflect the manufacturer's formal position (Simos, 1998).

2.1.3.1 Input Parameters, Geometry and Balance. Both existing and projected aircraft are modeled through basic parameters that can be assigned interactively, in any order. A full re-design procedure is executed automatically whenever a value is changed and if new output is requested. More than 200 possible aircraft input parameters are available, but most aircraft definitions typically require only 50 to 60 of these. Given these input parameters such as aspect ratio, wing area, design-cruise-mach or fuselage dimensions (the full list of possible input parameters is available in Appendix D), the PIANO system calculates all other necessary geometric data, wetted areas and volumes. Given basic wing specifications, PIANO will generate available internal fuel capacity and balance the design, locating the wing along the fuselage and sizing the tail areas according to statistically derived equations. The user can also move the wing location, tail areas and stretch the fuselage at will (Simos, 1998).

2.1.3.2 Aerodynamic Characteristics. PIANO calculates an aircraft's complete aerodynamic Lift-Drag Polar based on its geometric characteristics and allowing for various technology-level parameters. Detailed classical drag buildup techniques are used and the shape of this polar may be manually adjusted. High-speed compressibility drag and divergence Mach numbers are estimated using procedures previously developed by the Royal Aircraft Establishment (now DRA), allowing for

different levels of supercritical or conventional airfoil technology. PIANO evaluates low speed aerodynamics using a blend of textbook methods with a choice of commonly-used flap types. Factors on the estimated overall maximum Lift Coefficient, CL_{max}, and low-speed Lift Over Drag (L/D) ratios are often used to adjust these values which are sensitive to configuration details and may require wind tunnel or flight-test verification (Simos, 1998).

2.1.3.3 Mass Estimation. Each aircraft's mass characteristics are predicted using conventional preliminary design techniques. This is performed using a mixture of semi-empirical and semi-theoretical equations. The methods have been calibrated against industry-derived data, including component mass breakdowns that are not generally available in the public domain. All the calculated items can be individually factored or overridden by the user to exactly match known masses, or to simulate the use of advanced technology materials (Simos, 1998).

Each airframe structural component is assessed separately using one of several different estimation methods for the wing, fuselage and tail weights. Design load factors are evaluated according to standard FAR-25 rules with emphasis on wing weight prediction. The explicit, fundamental wing box weight equations are sensitive to all the major design parameters such as aspect ratio, sweepback, wing area, thickness/chord ratios, loading conditions. The Maximum Takeoff Weight (MTOW) can be either input directly, calculated iteratively to satisfy a range requirement or derived from a parametric study or optimization procedure (Simos, 1998).

2.1.3.4 Range and Flight Performance. Detailed flight performance and range evaluations are derived from first principles, based on the current aerodynamics,

mass, and engine characteristics. The climb, cruise and descent segments of the main mission all may be analyzed using detailed step-by-step techniques. The cruise phase may be flown at a constant altitude, over a sequence of altitudes using a step-up profile or using a continuously varying optimal altitude using a drift-up profile. The cruise Mach number can be determined to match various conditions such as high-speed, maximum Specific Air Range, economy or 99% of maximum Specific Air Range. Operating Ceilings and Initial Cruise Altitude Capability can be determined at various combinations of engine rating and residual rate of climb with all engines operating while engine-out cases may be evaluated to match a residual climb gradient (Simos, 1998).

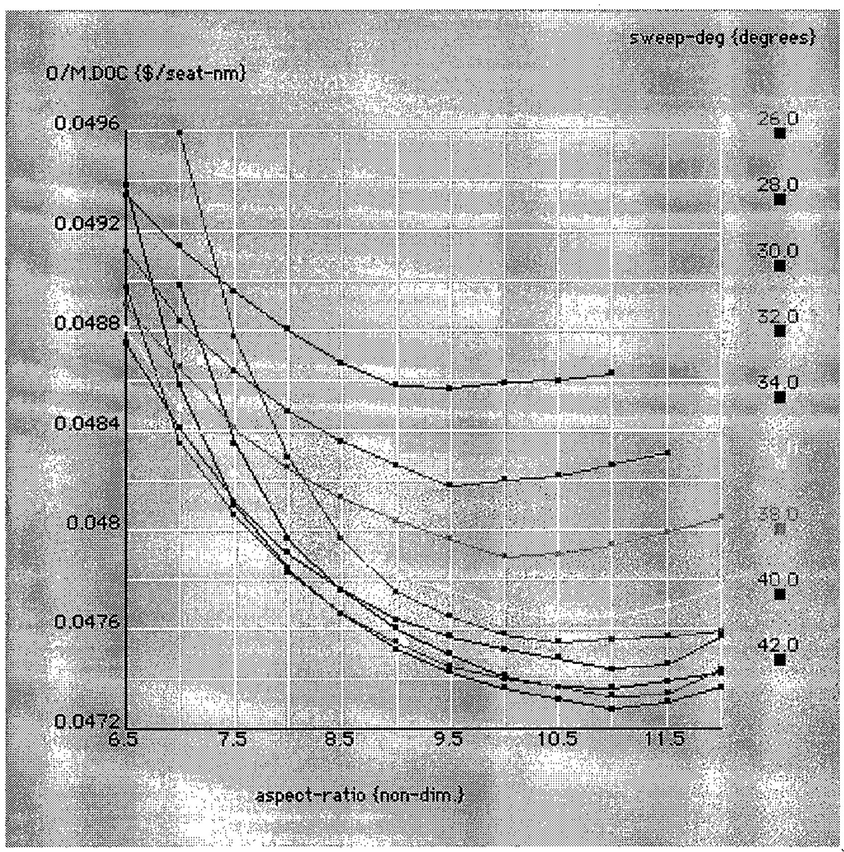
2.1.3.5 Takeoff and Landing Field Lengths. PIANO calculates the Takeoff Field Length (TOFL) and Landing Field Length (LFL) from first principles based on FAR-25 definitions. Engine-out during takeoff cases may be examined iteratively assuming different failure speeds to determine accelerate-go and accelerate-stop distances. The critical conditions and the corresponding Balanced Field Length (BFL) can then be determined. Takeoffs can be simulated at off-ISA (International Standard Atmosphere) conditions and non-zero field elevations. Landing Field Length is evaluated using similar principles (Simos, 1998).

2.1.3.6 Engine Modeling. Engine characteristics are modeled in terms of data matrices and can be scaled to any reference thrust. The maximum takeoff (MTO), maximum climb (MCL), maximum cruise (MCR) and maximum continuous (MCO) ratings are represented separately and vary with altitude and Mach number. Fuel flow or specific fuel consumption characteristics can be modeled in various ways using separate matrices for idle thrust and idle fuel flow. The current engine database within PIANO

consists of about 20 models representing various propulsive systems including turbofans, propfans and turboprops. PIANO comes with built-in facilities for the automatic non-dimensionalization of data. Smooth or linear data interpolation and extrapolation options are available within PIANO (Simos, 1998).

2.1.3.7 Parametric Studies and Optimization. A general facility within PIANO provides for conducting and plotting parametric studies using any two of the major design parameters. The results are saved in text files and can be read by other software packages such as Excel for further post-processing and plotting. Parametric studies are possible and are approximately equivalent to running multiple point designs (Simos, 1998). Figure 2.1.3.7 below illustrates a sample parametric study produced within PIANO.

Figure 2-3: Parametric Study Using PIANO



PIANO is an interactive tool and its ability to continually retain the decision-maker in the loop is crucial. It provides multivariate optimization features as an additional technique that may suggest potentially useful combinations of parameters. Aspect-ratio, sweep, wing-area and other basic parameters may be freely varied. It is possible to specify aircraft mass, fuel burn, or DOC (Direct Operating Cost) as the objective function to be minimized, subject to a variety of constraints such as field length, range and ceiling requirements. The numerical optimization methodology is based on the Nelder-Mead Sequential Simplex method modified to cater for inequality constraints (Simos, 1998).

2.1.3.8 Operating Costs and Emissions. Direct Operating Cost is calculated using a method derived by the Association of European Airlines. All the relevant parameters such as prices, amortization periods, interest rates, etc. may be user adjusted. Emissions of Atmospheric Pollutants (NOx, CO and HC) may be estimated over an entire flight profile based on a public-domain fuel-flow-based method developed by a major aircraft manufacturer (Simos, 1998). A complete conceptual aircraft design example with actual PIANO charts and output information may be found in Appendix D. PIANO was considered for use in the thesis effort but its exhorbitant cost was considered prohibitive.

2.1.4 Current Aircraft Conceptual Design Example: Boeing 777. The conceptual design of the Boeing 777 is widely considered to be a revolutionary change in the way the aircraft design business is performed. The 777 design team made unprecedented use of computer modeling to perform preliminary aircraft design steps which previously were expensively and painstakingly done using physical models. Geoffrey Fox, of the Northeast Parallel Architectures Center at Syracuse University emphasizes that high-performance computing is vital in every aspect of new aircraft design. He also points out that less than five percent of the initial costs of the Boeing 777 aircraft were incurred in computational fluid dynamics airflow simulations--the classic Grand Challenge in the field of aircraft design (Fox, 2000).

Ilan Kroo of the Stanford University Department of aeronautics and Astronautics mentions specifically how the Boeing 777 design team took advantage of the emerging field of Computation Based Design (CBD). CBD is called by several names: Computational Prototyping, Multidisciplinary Design and Optimization, or Simulation-Based Design. There are many interpretations of these terms, however, they all involve a

combination of simulation, modeling, and design tools used to design complex systems (Kroo, 1996).

Kroo also states that the Boeing 777 is an excellent example of the uses of computational prototyping. Remarkably, the 777 was designed, test flown, and repaired before a single component was manufactured. But this example also illustrates how little we are exploiting the potential of computation-based design. When the 777 team put together the first virtual airplane prototype, the decisions had been made that would lock in greater than 70% of its life-cycle cost. The wing planform shape for the 777 was designed mainly by the high speed aerodynamics group. Only heuristic considerations were given to low-speed performance or structures with essentially no input from the stability and control group. One of the goals of simulation-based design is to incorporate multidisciplinary and cross-functional requirements and objectives into the early stages of the design process. In these early stages is where computational prototypes, optimization and similar tools will make the biggest difference (Kroo, 1996).

2.1.4.1 EASY5 Design Software in Design of Boeing 777. Many different software packages have recently appeared to aid the systems engineering process. EASY5 is such a family of software tools used to model, simulate and analyze dynamic systems. EASY5 is an extension of the original EASY4 software designed by Boeing over twenty years ago. EASY5 is used to solve modeling, analysis and design problems in mechanical, electrical, aeronautic/astronautic systems, hydraulic, fluid power, pneumatic, thermal and control systems -- and for combinations of these systems. EASY5 can be used, and has been used, to solve problems in applications ranging from earth movers to spacecraft. The Boeing Company utilized EASY5 in the conceptual

design of the 777 passenger aircraft. EASY5 is currently maintained by the Mathematics and Engineering Analysis group of the Boeing Information & Support Services division of The Boeing Company (Boeing, 1999). EASY5 is publicly available through The Boeing Company and a free demonstration disk may be obtained at email address easy5.sales@boeing.com or by calling 1-800-426-1443 (or 425-865-6695), extension 4 (Boeing, 1999). EASY5 does not actually perform FEA for a model such as an aircraft, but is produced to easily interface with leading aircraft FEA software packages such as MSC.NASTRAN and many others.

No engineering analysis package performs well in every area, so the best way to model and analyze a complete system is to employ various types of engineering software. Very early on, Boeing recognized the need to integrate EASY5 to other leading computer aided engineering (CAE) tools. EASY5 possesses the ability to seamlessly link with other CAE tools. This gives EASY5 the capability to perform complete system prototyping (Boeing, 1999).

2.2 Blended Wing Body Studies

2.2.1 Blended-Wing-Body Aircraft Conceptual Design Study. This thesis effort was inspired by a NASA sponsored technology study performed by The Boeing Corporation under the Advanced Concepts for Aeronautics Program. Rapid construction, evaluation and change of aircraft design provide the motivation for our work. The thesis sponsors are extremely interested in developing a flexible, parametrically defined model that may be quickly evolved from a conventional, circular fuselage aircraft to a Blended Wing Body (BWB) design and subsequently evaluated.

Current transport aircraft design is embodied by the conventional cylindrical fuselage airframe. The blended wing body transport concept has become a hot topic when discussing future aircraft design. A NASA/Boeing study suggests that departing from the conventional cylindrical fuselage transport offers many advantages in aerodynamics, structures, human factors, systems, and economics. The idea has also been studied simultaneously by many organizations including McDonnell Douglas and several academic teams (NASA, 1997).

Since the 1950's, classical cylindrical or near cylindrical fuselage type subsonic jet aircraft have evolved along the same basic design: a circular fuselage mated to swept-back wings. As a result, performance and efficiency gains have been generally incremental. Substantial gains have been achieved to date based on the sum of these incremental gains. For this reason, present jet aircraft are far more economical and efficient than those in the past. However, in the present competitive aircraft design environment, still greater gains in efficiency and economy must be achieved to meet market requirements. NASA studies indicate that significant potential gains may be accomplished in aircraft design through further improvement of the conventional aircraft. However, fifty years of refinement on a good design concept must not exclude the investigation of other paradigms (NASA, 1997).

The Advanced Concepts for Aeronautics Program was established by NASA-Headquarters to investigate new paradigms in aeronautical concepts while working with industry and academia. One such concept was the Blended-Wing-Body (BWB) Technology Study (see Figure 2-4), a 3-year technology identification program to determine the technical and commercial viability of an advanced unconventional

subsonic aircraft. This unique transport concept addresses future NASA goals for emissions, noise, capacity, safety, and air travel cost (NASA, 1997).

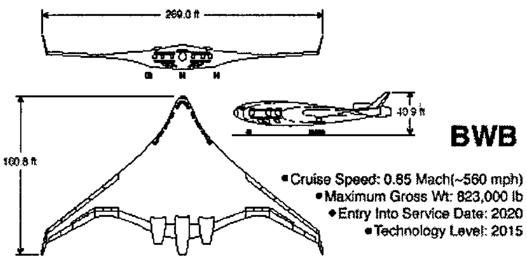


Figure 2-4: BWB 3 View

The report concludes that worldwide air travel passenger demand is expected to triple within the next 15 to 20 years. In the past, the number of aircraft, aircraft operations and passenger capacity have all increased to accommodate an increasing load of passengers. However, relatively few new airports are being constructed, and the current airspace operations system is becoming saturated. This trend makes larger aircraft more attractive to airline carriers. Larger aircraft have also been one of the airlines' main means of reducing operating costs. NASA is interested in large aircraft because of their ability to carry more passengers on fewer planes. This capability inherently reduces the cost per passenger mile, the number of required aircraft and emissions. In addition to passenger applications, civil and military cargo aircraft operators are also very interested in the built-in economy of scale that large transport aircraft concepts possess (NASA, 1997).

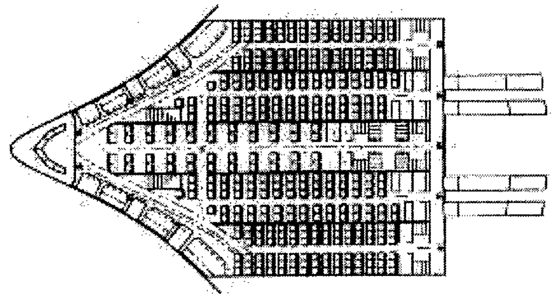


Figure 2-5: Upper Passenger Deck

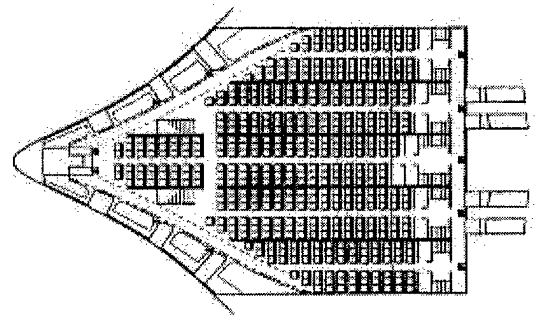


Figure 2-6: Lower Passenger Deck

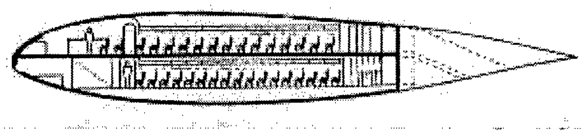


Figure 2-7: Passenger Deck Cross Section

The NASA BWB concept is designed to carry 800 passengers within a single lifting surface that integrates engines, wings, and a double decked cabin. The design utilizes technology levels expected for service in the 2020 timeframe. The results of several design studies indicate that the BWB would be efficient and economical relative to its conventional competitors. With a 7000+ mile range and a cruise speed of 560 mph (Mach 0.85), the BWB would reduce fuel burn and harmful emissions per passenger mile by almost a third compared with other modern aircraft. NASA concludes that the BWB shape possesses inherent aerodynamic and structural advantages. It has a low wetted area to volume ratio and reduces interference drag usually present where the fuselage meets wings and stabilizers. The structural sections are deep and efficient, while the blending of wing and fuselage yields a favorable wing span loading (NASA, 1997).

The BWB has many obstacles to overcome if it is to become a reality. In the BWB study, NASA states that concerns are the dynamics and control of flying wings,

pressurization of a non-circular fuselage, drag due to thickness, and engine intake of turbulent air. Rapidly advancing technology in the near future will undoubtedly supply solutions to these problems.

The BWB design specifications include an estimated takeoff gross weight of the aircraft is 823,000 pounds (composed of 75% composites and 25% metal), propelled by three 60,000-pound class turbofan engines. The engines are located on top of the wing, aft of the passenger compartment. NASA claims this design works very well for balance and has several beneficial side effects. Improved safety is possible because the turbines and compressors are completely clear of the fuel, pressurized compartments and main structural elements. The large fans on the high bypass-ratio engines are shielded from the ground by the center body, which improves noise characteristics for people on the ground. The BWB design compares favorably with modern large aircraft (see Figure 2-9 below). It has a 60-foot wider wingspan and a 70-foot shorter length than the Boeing 747-400, which carries about half as many passengers, weighs about 6-percent more, and uses four 60,000-pound class engines (NASA, 1997).

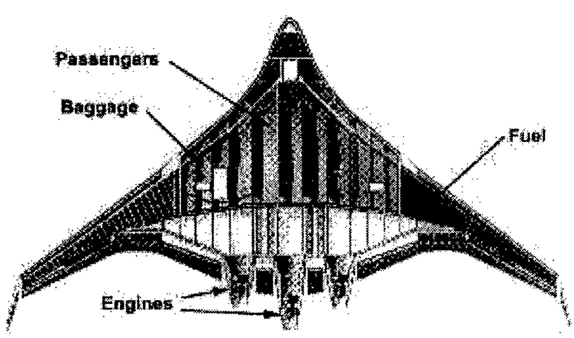


Figure 2-8: BWB Structure, Components.

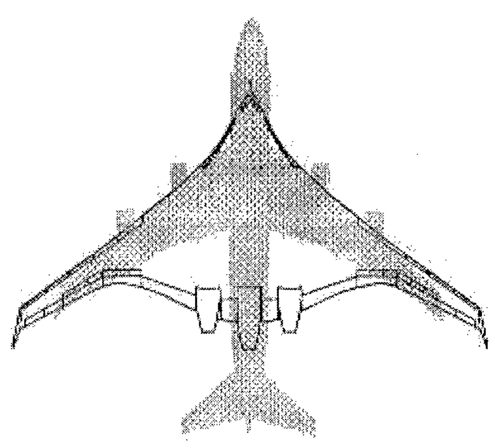


Figure 2-9: BWB vs. 747-400

Because of its extremely integrated design, NASA and Boeing utilized multidisciplinary design optimization (MDO) processes extensively to address technical issues in configuration design, aerodynamics, structures, propulsion, and flight mechanics. NASA states that analyses of the BWB configuration indicates significant cost and performance benefits over projected conventional concepts using equivalent 2020 technologies: 21-percent increase in lift-to-drag ratio, 17-percent decrease in fuel consumption, 6-percent decrease in maximum takeoff weight, as well as a 12-percent decrease in operating costs (NASA, 1997).

The BWB Technology Study included extensive performance and weights analyses at the conceptual design level and provides an excellent example of modern aircraft conceptual design. Computer aircraft models were generated then analyzed, and the process was iterated until all constraints were met. Cost and manufacturing models were of great importance in this study. Emissions and cost reduction characteristics were the primary metrics in assessing the design. Basic structural concepts were examined, particularly for the pressurized passenger cabin, then a global finite element model (FEM) was developed. The FEM was used to determine overall structural load paths, complete basic structural sizing, compute aeroelastic stability derivatives, and check initial centerbody and wing weights. Computational Fluid Dynamics (CFD) models were created and analyzed to find the maximum cruise speed for the thick airfoil and to examine stability and control derivatives. High speed wind tunnel tests were conducted to provide confidence in the CFD modeling results and verify the predicted cruise Mach number (Figure 2-10). Low-speed wind tunnel tests to determine stability derivatives and identity possible handling quality deficiencies were conducted with an

11.5-foot wingspan (4-percent scale) model (see Figure 2-11). Flight tests were also conducted by Stanford University (see Figure 2-12) on an instrumented 17-foot wingspan (6-percent scale) radio-controlled model of the BWB to study flight control options and verify low-speed stability and control derivatives (NASA, 1997).

Figure 2-10: High-Speed Model

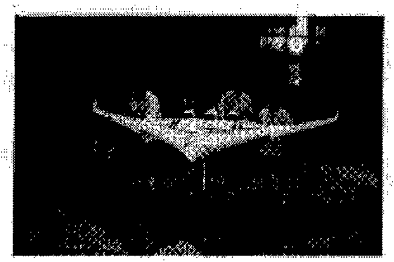


Figure 2-11: Low-Speed Model

Figure 2-12: 17-Foot Flying Model

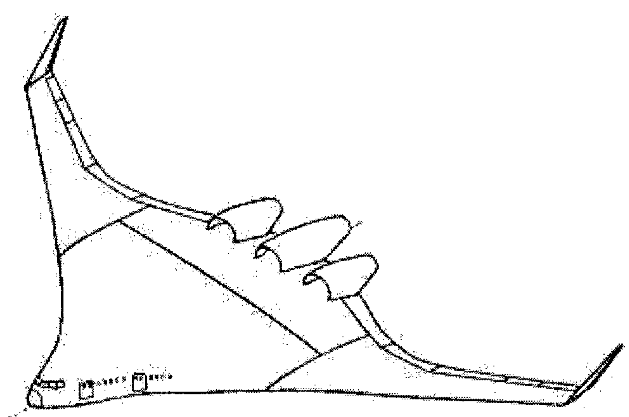


Figure 2-13: BWB Isometric View

The BWB team successfully demonstrated that NASA, industry, and academia can work together to design and test unique aircraft concepts that promise large leaps in subsonic transport efficiency. NASA and its academic and industrial partners are currently overseeing efforts to further address critical technologies to BWB development. A conceptual view of the projected life size BWB is illustrated in Figure 2-13 above. Further evolution of the BWB design should be pursued in the future race to improve air transportation.

2.2.2 Cranfield College Of Aeronautics Aircraft Concept Study. For the past 80 years, aeronautics has been devoted to refining the most efficient aircraft designs and squeezing out every last drop of performance from formerly existing aircraft design studies. Improvements are made in small increments and are expensive and difficult. A more radical approach is now required to meet the great demand of airlines around the world (Cranfield, 1999).

The Cranfield College of Aeronautics is studying cutting edge aircraft design technology to explore new ways to configure and develop design tools. The Blended Wing Body aircraft concept is a new worthy design that has many advantages in compared to the previous design technology in the following areas (Cranfield, 1999:14):

Potential Advantages of BWB:

Aerodynamics
Low wetted area to volume ratio
Form conducive to low interference drag
Structure
Efficient deep sections
Favorable span loading
Human Factors
Huge volumetric capacity
Flexible cabin layout potential
Systems

Potential for highly integrated airframe/engine
Ideal configuration for application of laminar flow technology
Significant advantages from control configuring the vehicle
Economics
Particularly suitable for high capacity applications
Significant Direct Operating Cost reduction should be achievable

2.2.2.1 Cranfield College of Aeronautics BWB Program. Currently aircraft design is a relatively easy and straightforward process because there are readily available, proficient engineers experienced in applying design tools to previous aircraft designs. However, for the latest aircraft design processes and concepts, there are no proficient experts to call upon. For this reason, an infrastructure of tools and procedures must be established in advance to make a new design environment possible (Cranfield, 1999).

The Cranfield College of Aeronautics expended 75,250 man-hours, including a 12,000 man-hour flight demo, a 52,000 man-hour preliminary design, and 11,250 man-hours of support to design a BWB aircraft which was similar in payload and mission performance to the A3XX-200 aircraft, yet with superior direct operating costs. The Cranfield study was to meet the following objectives:

> "To complete a detailed design study of a fully optimized BWB configuration with integrated propulsion system, incorporating all appropriate technologies (e.g. laminar flow) within a rigorous framework of constraints to ensure that it can be successfully and profitably manufactured and operated and to the benefit of passenger appeal and safety. This will provide a considerable degree of confidence that all major design problems have been identified and addressed."

Many milestones must be accomplished to develop this new design concept:

The creation and continued development of design tools

Development of appropriate design methodologies

An incremental program of detailed design studies

The design and manufacture of a sub-scale flying demonstrator

Detailed studies within a number of identified Key Technology Areas feeding back into the tools and the methodologies.

(Cranfield, 1999)

2.2.2.2 Baseline Concept Preliminary Design. The BWB program has four main phases: a baseline preliminary design, an advanced technology preliminary design, a sub-scale flying demonstrator and a series of supporting studies that occur simultaneously with the rest of the program. The baseline preliminary design phase began by analyzing a new design concept using the pre-existing AIRBUS A-3XX-200 high capacity civil airliner study. The Airbus design will address airport operational constraints and will assumes a technology level consistent with A3XX. The study will proceed to a level of detail sufficient to resolve potential structural and system problems and explore solutions to those problems. This level of detail is necessary since many of the human factors and engineering challenges will not be apparent at the conceptual design stage (Cranfield, 1999).

2.2.2.3 Advanced Concept Preliminary Design. The advanced technology concept preliminary design study will build on the baseline study. It will incorporate a number of synergistic technologies that will enable realization of the full potential of the BWB concept. The basic A-3XX-200 specification will be followed in conjunction with airport operational constraints to ensure that a direct comparison will be made between the AIRBUS A-3XX-200 and the baseline BWB design. The baseline BWB design will

be evaluated using in-depth structural finite element analysis. Technologies likely to be incorporated include Hybrid Laminar Flow Control (HLFC), a Stability Augmentation System (SAS) and an advanced propulsion system (Cranfield, 1999:22). The Cranfield University study demonstrates the worldwide commitment to the advantages which a blended wing body aircraft design may bring to the aircraft industry.

2.3 Multidisciplinary Optimization

Most design problems have multiple criteria for "success" and each criterion should be met for a "successful" design. For instance, a cargo aircraft might be required to be capable of flying a certain payload a certain distance. Often, the criteria are at odds with one another; a “successful” car design may be one which is simultaneously inexpensive and one which has lots of features, when features cost extra money. A simple solution to the problem of criteria at cross-purposes is to divide the design problem at the beginning of the design process and set constraints that each team must meet. In classical aircraft design, one team may be assigned to develop a fuselage that can carry a particular payload, while another team may be assigned to design a wing that would provide the necessary lift for that particular payload. When the teams have completed their separately-designed components, they are integrated into a complete design, which should (ideally) meet the original requirements and constraints.

Multidisciplinary optimization (MDO) in the conceptual design process brings the integration of various subsystems up one level. Instead of a team optimizing wing design while another team optimizes fuselage layout, and then fusing their designs together, the entire plane is optimized for wing design and fuselage layout at the same time. The

results of changes performed on one subsystem can be seen both in that subsystem and all other subsystems.

As computing power has become "cheaper" (both in time and cost), analysis programs have become more robust and able to accept more complex and detailed designs. It has become feasible to tie optimization routines into analysis codes. MDO represents the next step in creating systems of analysis/optimization routines. MDO relies on computing power to evaluate all of a design, instead of dividing the design into separate areas and optimizing each area.

Often, the benefit of MDO is its ability to provide insight into the coupling of subsystems. A particular advantage of using MDO early in the design process is to explore areas of the design space to see where coupling affects the design. The Junkers designs of the 1930's and Northrop's "flying wing" designs of the 1940's are examples of the positive coupling of subsystems. The flying wing melded the fuselage into the wing, creating a heavier wing, but one which incorporated the fuselage's cargo-carrying capability, negating the need for a "fuselage" as such. The flying wing design could not come about from a classical aircraft design approach, which, through its division of the aircraft into "fuselage" and "wing" systems, mandates the existence of separate fuselages and wings.

Chen and Lewis propose a design system in which a design team is broken down into single-discipline teams, but the results of each team's analysis are available to each team. Teams reevaluate their designs based on other team's information, and a more robust design develops as iterations take place (Chen and Lewis, 1999:983).

MDO can be used on designs of varying complexity at various stages in the design process. MDO can be used in a general "system level study" to explore combinations of options, each combination being a potential solution to the study problem (Rowell and others, 1998). Rowell's group describes using a multidisciplinary approach to evaluating concepts for a space transportation system.

MDO can be used in conceptual design of hardware, where the number of design parameters is limited (Sevant and others, 1999). Eleven design variables are used to describe the geometry of a proposed high speed civil transport. Constraints to the optimization problem of minimizing drag are formed by considering other objectives which the design needs to satisfy.

The "optimization" in MDO can take place in a number of ways. If a quantitative value hierarchy has been established with corresponding scoring functions and weights for each criterion, given designs can be evaluated by this one, global fitness function. The fitness function evaluates (instead of optimizing) each design. This type of optimization is often valid in high-level conceptual design studies, in cases where the number of criteria is small, or where the number of possible alternatives is limited, so that the members of the feasible set can be analyzed in a short amount of time.

If each criteria requires its own optimization program, a global fitness function can still use the outputs from each optimization routine to create a quantitative score for each design that was considered. Again, this approach assumes that designs are created outside the analysis/optimization program, and the program itself doesn't create new designs or improve on them; it simply determines the fitness of the given design.

If there is no global fitness function, the design space can still be characterized by use of Pareto analysis or any number of optimization techniques that can be applied to the vector of objective functions with an intent to find the "optimum" design based on a sub-optimum starting point. Rajadas and others describe a method which combines the Kreisselmeier-Steinhauser function approach (which combines multiple constrained objective functions into one unconstrained composite function) with the ability to weight the importance of the component objective functions (Rajadas and others, 1997:829). A point raised in several papers applies to any potentially non-linear optimization problem: most optimization procedures converge to a local optimum, and require higher level intervention to be applied to find a global optimum, either through the use of heuristics or intervention of the end user.

Alternately, MDO concepts can be used without an eye towards analytical optimization. Bishop et al. use MDO techniques to see how switching components of their aircraft wing design affected the weight of the design when different constraints (such as loading due to flutter and roll) were imposed. Based on the results, the authors gathered more information about how loading affected the particular wing structure being studied, while not specifically determining an overarching, global optimum design suitable for all of the conditions they considered (Bishop and others, 1997).

2.4 Aircraft Life Cycle Cost Modeling

The total Life Cycle Cost (LCC) of an aircraft is defined as the expense to acquire, operate and dispose of an aircraft. The LCC of an aircraft historically includes total program cost for acquisition of airframe, avionics and engines, operations and

support (O&S), and disposal costs. LCC analysis is a discipline that is extremely important during conceptual aircraft design. Aircraft are typically defined in terms of required performance. However, the customer, or aircraft purchaser, must use some criteria other than aircraft performance to select the best proposal. While there may be some differences in technical credibility, data substantiation, and intrinsic design qualities, the final contractor selection will probably hinge on cost. The United States Air Force, as well as the majority of the Department of Defense, has been forced to face extreme funding reductions since the late 1980's. The focus has shifted from meeting set performance goals at any cost to meeting set cost goals by modifying or reducing performance requirements. The Air Force must now closely consider not only the initial price of purchasing an aircraft, but the entire LCC of that system. Similarly, civilian aircraft operators are faced with increasing financial pressures. While civilian air transport is cyclical, financing an airline is not. In an effort to conserve capital, airlines have extended their planning horizons so that LCC, not financing cost, is the cost number to which attention is paid.

2.4.1 Elements of Life Cycle Cost. Noted author Daniel Raymer provides some additional details on aircraft life cycle cost (LCC). Figure 2-14 below displays the elements of LCC. The box sizes are roughly proportional to the magnitude of the typical aircraft costs. RDT&E, or research, development, test, and evaluation includes technology research, design engineering, prototype fabrication, flight and ground testing, and evaluations for operational suitability. Aircraft conceptual design cost is included in the RDT&E cost. Certification cost is included under RDT&E for civil aircraft. For military aircraft, RDT&E includes the airworthiness demonstration cost, mission

capability, and compliance with Mil-Specs. RDT&E costs are fixed regardless of how many aircraft are produced (Raymer, 1990).

The aircraft flyaway, or production, cost covers the labor and material costs to manufacture the aircraft including airframe, engines, and avionics. This cost includes production-tooling costs as well as manufacturer's overhead and administrative expenses. Production costs are recurring and based on the number of aircraft produced. The cost per aircraft decreases as the number of aircraft produced increases due to the learning curve effect (Raymer, 1989).

The purchase price of a civil aircraft roughly equals the RDT&E and production costs, plus a fair profit. Because the RDT&E costs are held constant, one must assume a quantity of aircraft produced to determine how much of the RDT&E costs each sale must recover. For military aircraft, the government directly pays the RDT&E costs in the appropriate life cycle phase so these costs need not be recovered during production. Military-aircraft acquisition, or procurement, cost includes production costs, costs for initial operational deployment spares as well as ground support equipment costs such as flight simulators and test equipment. For civil aircraft, these are normally purchased separately. Military program cost includes the total cost to develop and deploy an aircraft into the military inventory (see Figure 2-14). Some aircraft such as the B-2 stealth bomber require special ground facilities for operational deployment. Cost sharing is a recent trend in military aircraft where the contractor is invited to share some RDT&E costs with the expectation of recovering them later in production. It is not yet clear whether future administrations will permit full cost recovery in the future (Raymer, 1989).

After an aircraft is delivered to an operational unit or customer, the aircraft enters the O&S phase. In the military environment O&S costs indicate an organization's commitment to military readiness. Readiness is a measure of the degree to which a certain force structure such as an aircraft wing has been activated by O&S expenditures (Hildebrandt, 1990).

O&S costs are usually much greater than development and production costs because they are incurred over the long lifetime of the aircraft. In the current environment of aircraft modification and improvement programs, many Air Force aircraft such as the B-52 bomber are being utilized far beyond their original projected lifetimes. Greg Hildebrandt and Man-bing Sze note that in military applications, O&S costs cover fuel, oil, aircrew, maintenance, and other various indirect costs such as personnel support, training ordnance and replenishment spares (Hildebrandt, 1990). For civil aircraft, insurance is also included in the cost of operations. In commercial aircraft operations, the depreciation of an aircraft based upon purchase price is also considered a part of operating costs. Depreciation refers to the gradual reduction of the purchase value over a number of years according to a certain schedule. The simplest schedule of depreciation is a straight-line formula, in which each year's depreciation is the purchase price divided by the total number of depreciation years. Commercial aircraft are usually depreciated over only 12-14 years, although they may have a useful life of greater than 20 years (Raymer, 1989).

The final element of life-cycle cost is the disposal phase. Obsolete military aircraft are flown one final time to Davis-Monthan Air Force Base, Arizona for mothballing and storage. This expense is not relatively large, so it is frequently ignored

in LCC estimation. Civil aircraft have a negative disposal cost because have monetary value (typically 10 % of purchase price) in the resale market (Raymer, 1989).

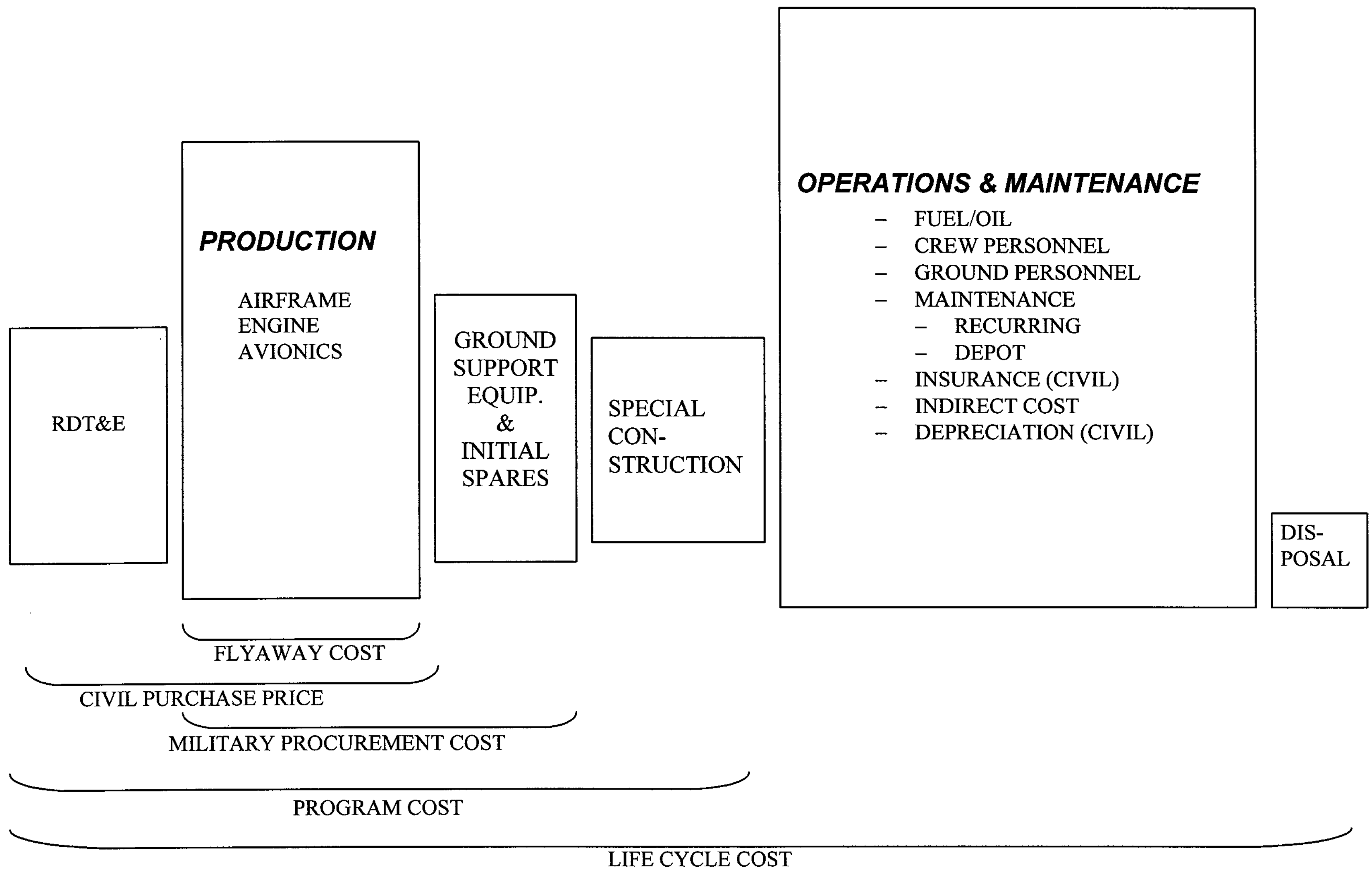


Figure 2-14: Elements Of Life Cycle Cost (Raymer, 1989).

2.4.2 Cranfield BWB Study Cost Estimate. The Direct Operating Costs (DOC's) of an aircraft are highly dependent on the initial acquisition cost because the airline experiences aircraft depreciation and finance payments. For this reason, the aircraft acquisition cost must be estimated. Two forms of analysis were carried out under the BWB study performed by Cranfield University's Aeronautical Engineering Department. The conceptual BWB aircraft design was known as the BW-99. The top-down approach plotted the unit acquisition costs against a single independent variable which represented an aircraft characteristic. The trend is then approximated by a function using statistical methods. The bottom-up approach uses statistical data to estimate the cost components of an aircraft project. The cost components are then summed to obtain the overall project cost, which can be divided by the number of aircraft to yield the cost per aircraft. Figure 2-15 below demonstrates how unit acquisition cost varies with aircraft build numbers:

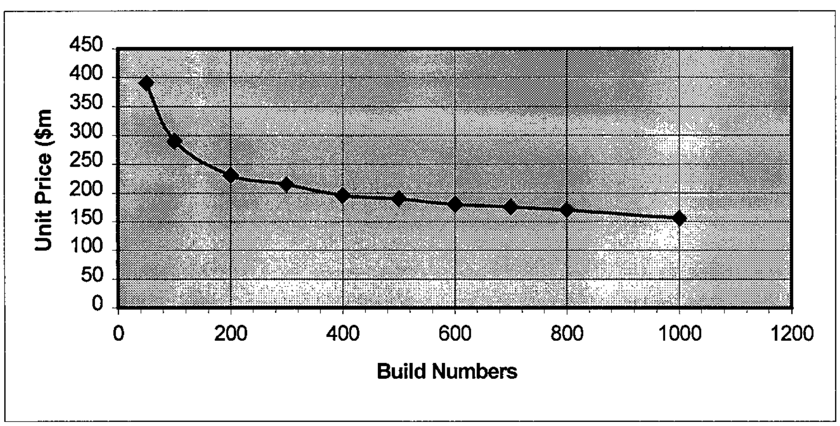


Figure 2-15: Unit Price vs Number of Aircraft Produced

The various cost-estimation methods used for unit acquisition cost pointed to a cost of $164M per BW-99 for a production run of 100 aircraft. The cost calculation demonstrated that the unit acquisition price was highly dependent on the price of engines

and avionics. Engines and avionics each represent about 20% of the acquisition price. This means that to accurately estimate aircraft price, the cost of the engines and avionics must be estimated as well (Cranfield, 1998).

The DOC's of the BW-99 were compared with its largest existing competitor, the Boeing B747-400 using the AEA (Association of European Airlines) method. When the unit price of the BW-99 is assumed to be \$164M, it realizes a savings of 19% over the B747-400 in terms of DOC per seat mile. Even if the BW-99 unit price is assumed to be \$200M, a savings of 10% over the B747-400 is realized. Thus, it can be concluded that over a reasonable range of aircraft acquisition cost, the BW-99 represents a saving of 10-19% in direct operating costs when compared to the B747-400. Using AEA definitions, the breakdown of the BW-99 DOCs are as follows: (Cranfield, 1998)

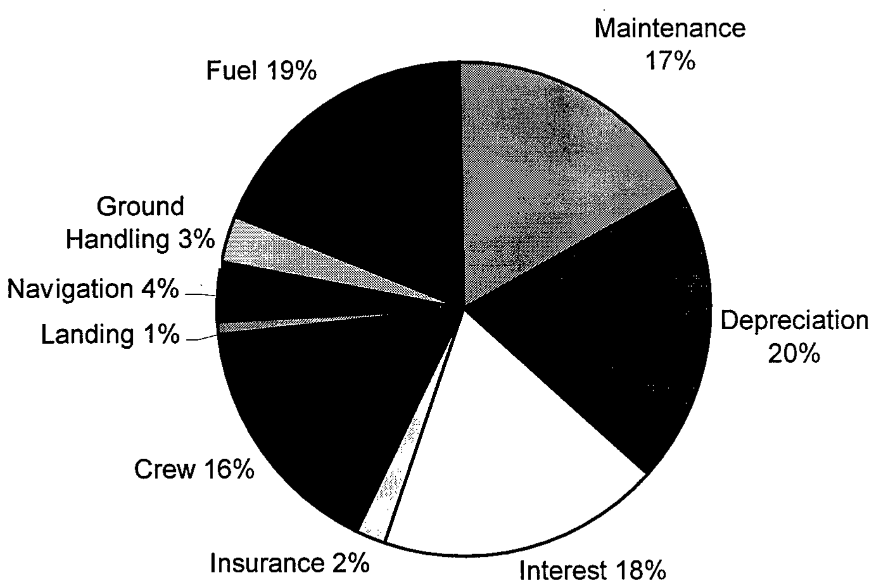


Figure 2-16: BW-99 Direct Operating Costs

2.4.3 Problems with Aircraft LCC Estimation. Aircraft cost estimation generally occurs in the fuzzy gray area between science, art, and politics. Cost estimation is mainly statistical and the final analysis of new aircraft cost will be based on the actual costs of previously existing similar aircraft. However, it is often extremely difficult to quantify the actual cost of a prior aircraft in terms meaningful to the planned aircraft (Raymer, 1989).

A majority of military aircraft production programs are extended over multiple fiscal years for political reasons. To reduce the current year defense budget the number of aircraft produced per year may be far below the optimal production rate. Production rates can be less than one per month. This greatly increases the unit cost per aircraft and ensures a cost overrun as the new aircraft production rate is decreased below its original planned value (Raymer, 1989).

It is very difficult to compare costs for two aircraft that are in production. The type of funding applied is also a source of problems. Program cost-comparisons can be made in then-year or constant-year dollars. Then-year dollars are actual dollars spent in each program year, past, present, and future. The inflation rate must be estimated in order to form future program cost estimates. Program costs should be compared using constant-year dollars, the actual dollars spent, ratioed by inflation factors to a selected year. Constant year dollars should also be used to establish a cost baseline for new aircraft cost-prediction. However, Congressional budgets and most other cost data are prepared using then-year dollars (Raymer, 1989).

Aircraft production rates and quantities pose another problem in cost comparison. As the number of aircraft produced increases, the manufacturer's knowledge grows and

the cost of subsequent aircraft decreases. This is known as the "learning curve" effect. When the production quantity is doubled the labor cost per aircraft decreases approximately twenty percent. This represents an eighty percent learning curve. Aircraft production typically follows a seventy-five to eighty-five percent learning curve. Because of the learning-curve effect, cost comparisons are not meaningful between a new aircraft entering production and an old aircraft already produced in great numbers. One final problem in cost comparison is that different costs are used by different organizations, often without proper identification. Comparing the flyaway cost of one aircraft to the program or life-cycle cost of another is worthless Costs must be properly identified before comparison to ensure uniformity (Raymer, 1989).

2.4.4 Aircraft Cost Estimating Relationships. Accurate estimation of future costs has also become extremely important in the acquisition of aircraft because modern military fighter, bomber and transport aircraft are some of the most expensive items purchased by the USAF. Acquisition of such financially significant items necessitates early modeling and tracking of total costs. Even during preliminary conceptual aircraft design it is possible to construct increasingly accurate models of aircraft life cycle cost. Parametric cost analysis for conceptual aircraft is possible using comparison by analogy with historical costs for other military aircraft. A linear regression of costs for recent military aircraft is formed using a database of historical information on similar, previously existing aircraft. Cost Estimating Relationships (CER's) are subsequently developed from a pool of potentially significant factors and tested for accuracy. The relationships may then be applied to conceptual aircraft using corrective factors based on the new aircraft's level of technological advance, initial operational capability date, etc.

The search for a reliable military aircraft cost estimating relationship (CER) began with an aircraft design text by Daniel Raymer. The CER found in the text was very simple and was applicable to all aircraft in general. It was based on an equation composed of the 70% of the empty weight of the aircraft and a unit cost multiplier of between $150 and $300 per pound. The Team added to this estimated cost a technical difficulty factor of 75% for the Raymer estimate applied to a BWB design. This provided an initial rough cost estimate for the preliminary aircraft design phase (Raymer, 1989: 495).

Experts were sought out and consulted from the C-17 cargo aircraft program office within the Air Force Materiel Command's Aeronautical Systems Center at Wright Patterson AFB, Ohio. Direct comparison by analogy with the C-5A Galaxy, the Air Force's current heavy lift aircraft, was suggested as a rough order of magnitude cost estimate. Assuming the new heavy lift aircraft design is a blended wing body design and approximately 75% more complex than the current C-5A design, then a direct estimate by analogy yields 1.75 times the cost of a current C-5A aircraft. This estimate by direct analogy provides a first "guess" of aircraft cost, but is very subjective in estimation of aircraft complexity compared to an existing aircraft (Bickel, 2000). More detailed cost estimates are also possible by using linear regression "trends" in historical aircraft costs.

The above discussions with C-17 Systems Program Office personnel pointed to a series of RAND studies begun in 1975. The studies were sponsored by the office of the Assistant Secretary of Defense (Program Analysis and Evaluation) as part of a research program focused on improved methods of estimating the development, procurement and operating costs of new weapon systems. The purpose of the studies was to derive

equations for estimating the acquisition cost of aircraft airframes as well as operations and support (O&S) costs. Hess and Romanoff state:

> Parametric models for estimating aircraft airframe acquisition, avionics, operations and support costs have been used extensively in government advanced planning studies and contractor proposal validation. These models are designed to be applied when very little is known about an aircraft design or when a readily applied validity and consistency check of detailed cost estimates is necessary. They require inputs that: (a) will provide relatively accurate results; (b) are logically related to cost; and (c) can easily be projected before actual design and development. The intent is to generate estimates that include the cost of program delays, engineering changes, data requirements, and inefficiencies of all kinds that occur in a normal program (Hess, 1987b).

Hess and Romanoff also state that such equations were intended primarily for use in long range planning specifically for military aircraft and not for contract negotiation or financial management. This means the CER's generated in the series of RAND reports are appropriate for use in this effort to estimate approximate acquisition costs for an aircraft model.

The first study attempted to develop a set of equations suitable for estimating the acquisition costs of military bomber/transport airframes in the absence of detailed design and manufacturing information (Hess, 1987b). This type of CER specific to USAF bombers and transports would prove useful in this effort for estimating the cost of a military transport aircraft. However, Hess and Romanoff concluded that a single acceptable estimating relationship for bomber transport aircraft could not be found. The study instead refers to a sister RAND study and recommends estimating costs for proposed bomber/transport aircraft based on the equation set developed for all mission type aircraft (Hess, 1987a). The RAND cost estimating relationship was:

$$T = 2.57*EW^{0.798}*MA^{0.736}$$

Where EW is the empty weight of aircraft in pounds, MA is the maximum airspeed of the aircraft in knots and T = total airframe cost for 100 aircraft in Fiscal 1977 Constant Year (FY 77) dollars. This cost estimating relationship was used to calculate an aircraft airframe cost estimate in section 3.6.3.3.

2.5 Systems Engineering Approach

2.5.1 Introduction. The systems engineering approach followed by the Team was based on approaches established in the past. The Team searched the literature for guidance on what sort of methodology to follow in generating a new aircraft design process as well as generating a new aircraft design itself.

2.5.1.1 Systems Engineering Framework. Based on the identified top-level needs, it was clear the problem was multi-faceted. Given such a problem, how does one simultaneously evaluate all concerns that need to be considered? Given several alternative solutions, how is the "best" alternative selected? How can one make sure that no critical aspects of the system are overlooked? The ability to find answers to these questions, and others, forms the foundation of the field known as *systems engineering*. As an initial step in the design of Blended Wing Body, the design team researched several works on the theory and practice of systems design using the systems engineering approach. Study in the systems approach gave the team insight into multidiscipline design challenges, development of a structured problem-solving methodology, breakdown of lifecycle phases, and incorporation of systems engineering tools, modeling techniques, and decision making methods.

The *systems approach* "represents a broad-based systematic approach to problems that may be interdisciplinary. It is particularly useful when problems are complex and affected by many factors, and it entails the creation of a problem model that corresponds as closely as possible in some sense to reality. Its usefulness increases with problem complexity because it permits the engineer to take a broad overall view of the problem under consideration. Thus a clearer understanding of constraints, alternatives, and consequences that are associated with the problem may be attained" (Hall, 1985). This summary of the systems approach clearly shows its relevance to the Aircraft Design Problem, being interdisciplinary and complex in nature. This theme is further emphasized by Sage as he states, "The systems engineering approach to problem solving emphasizes interactions and interrelations among the diverse parts of a problem" (McGraw-Hill, 1977).

2.5.1.2 Systems Engineering Definition. The top-down, interdisciplinary, and iterative aspects of systems design are evident in the following systems engineering definitions:

> Broadly defined, system engineering is 'the effective application of scientific and engineering efforts to transform an operational need into a defined system configuration through the top-down iterative process of requirements definition, functional analysis, synthesis, optimization, design, test, and evaluation.' The system engineering process, in its evolving of functional detail and design requirements, has at its goal the achievement of the proper balance between operational (i.e., performance), economic, and logistics factors (Blanchard, 1991).
>
> Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem: operations, performance, test, manufacturing, cost and schedule, training and support, and disposal. Systems Engineering integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation.

> Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs. (International Council on Systems Engineering, http://www.incose.org/whatis.html)
>
> Systems engineers are, of necessity, technical generalists. Systems engineering . . . is not intrinsically mathematical. Rather, it is organizational, judgmental, logical, goal-oriented, and admittedly must often be subjective (Beam, 1990).
>
> Systems engineering is the systematic application of proven standards, procedures, and tools to the technical organization, control, and establishment of: system requirements, system design, system management, system fabrication, system integration, system testing, and system logistics support (Reilly, 1993).
>
> Systems engineering is a branch of engineering that 'concentrates on the design and application of the whole as distinct from the parts... looking at a problem in its entirety, taking into account all the facets and all the variables and relating the social to the technological aspects.' (Ramo 1973)
>
> Systems Engineering basically consists of three elements:
> a) Systems Engineering Management plans, organizes, controls and directs the technical development of a system or its products.
> b) Requirements and Architecture Definition defines the technical requirements based on the stakeholder requirements, defines a structure (or an architecture) for the system components, and allocates these requiremenets to the components of this architecture.
> c) System Integration and Verification integrates the components of the architecture at teach level of the architecture and verifies that the requirements for those components are met (Martin, 1997).

2.5.1.3 Systems Architecting vs. Systems Engineering. What is *systems architecting* and how does it differ from *systems engineering*? As described previously, systems engineering encompasses the tools and methodology necessary to move from conceptualization to system implementation, with emphasis on the system as a whole and user needs. Indeed, systems architecting is also concerned with these same issues, and is occasionally used interchangeably with systems engineering. However, there are subtle, yet significant, differences between these systematic views of design.

Webster's Dictionary defines architecture as "the art or science of building." Traditionally, *architecture* refers to the planning and building of structures related to civil, military, or naval applications. In the last thirty years or so, the term has been applied to technical systems with increasing regularity, thus the common use of the terms *software architecture*, *computer architecture*, and the like. As stated by Rechtin (Rechtin, 1991), "The essence of architecting is structuring." Thus, the essence of systems architecting is structuring the system – "to bring structure in the form of systems to an inherently ill-structured unbounded world of human needs, technology, economics, politics, engineering, and industrial practice" (Hall, 1991). Clearly, this definition of architecting overlaps that of systems engineering. Rechtin identifies two areas in which distinctions are particularly important – function versus form and complexity versus specificity (Rechtin, 1991).

The guiding principle "form follows function" is basic to architecting, which focuses on the top-down design driven by *function* as opposed to *form*. Hillaker is quoted by Rechtin as stating (Rechtin, 1991), "System engineering is form-based and system architecting is function-based." With respect to complexity, the architect is "a specialist in reducing complexity, uncertainty and ambiguity to workable concepts. The systems engineer, in contrast, is the master of making feasible concepts work." (Rechtin, 1991) It follows that systems architecting "concentrate[s] on concepts, synthesis, top-level specifications, nontechnical as well as technical interfaces, and mission success", whereas systems engineering "concentrate[s] on defined subsystem interfaces, analysis, and performance to specification." (Rechtin, 1991).

The architect's role is most visible in the early stages of a design, when concepts are explored, both innovative and adaptive in nature. Beam describes architecture as "a matter of repetition among members of the class, and often repetition within a single member" (Rechtin, 1991) illustrating the adaptive nature of architecting, wherein functions are addressed by exploring how other systems are designed regardless of their form. For example, a variable geometry wing designed to provide the lift function for an aircraft may incorporate techniques borrowed from biological systems, in which electrical impulses cause muscle contractions. Although a wing and a human muscle are very different in form, the function of altering physical characteristics may be similar. As the design progresses, the visibility of the systems engineer increases, as the proposed concepts are refined, detailed, and implemented. With a system concept already suggested, the tools of systems engineering can most efficiently be brought to bear.

Why are the distinctions between systems architecting and systems engineering relevant to the Aircraft Design Problem? This design progressed from initial identification of needs and concept development, through the actual integration of subsystem components, necessitating an understanding of both systems architecting and systems engineering tools and techniques. Thus, the team performed both architecting as well as engineering roles. The line between these roles is indeed blurred, as the "architect hat" and "engineer hat" are sometimes worn simultaneously, especially during concept exploration and preliminary design. Once the design became more and more refined, systems engineering tools were more readily implemented.

2.5.1.4 Systems Engineering Dimensions. Hall divides the systems engineering approach into a three-axis morphological box, as shown in Figure 2-17. The

time dimension of systems engineering refers to the system lifecycle -- the sequences or phases that extend from initial conceptualization through system retirement. The *logic* dimension refers to the problem-solving process – the steps necessary to move the design from one lifecycle phase to the next. Finally, the *knowledge* dimension refers to the specialized knowledge from various fields necessary to address and solve the problems at hand.

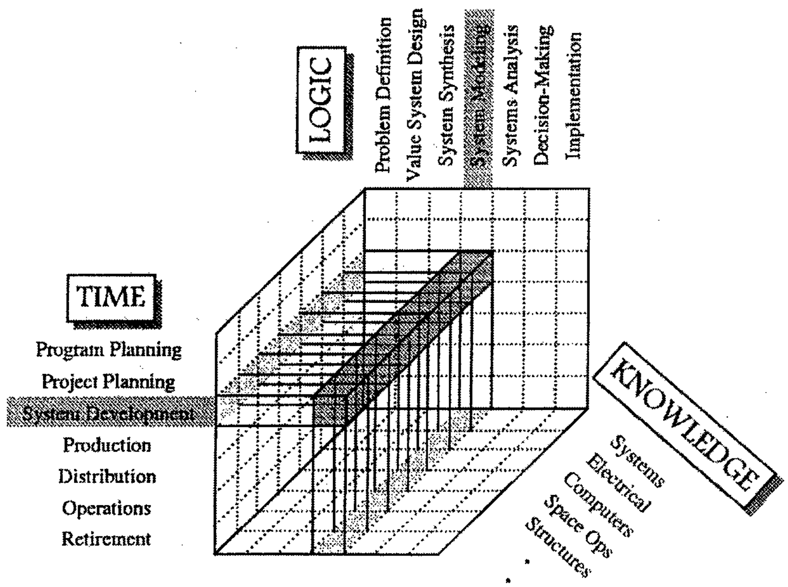


Figure 2-17: Systems Engineering Dimensions

2.5.2 Systems Engineering Process

2.5.2.1 Problem-Solving Methodology. As described in Section 1.3, there is an underlying process in a well-planned design which facilitates the evolution of the design from a problem statement to conceptual alternative solutions, and finally to a resultant design ready for implementation. The design process by which this evolution occurs can be a considerable design problem in and of itself. A process which encumbers

the design team and impedes conceptual evolution is not desirable. This situation can occur when a process is too rigid for the problem at hand. An overly rigid process can lead to overemphasis of process objectives at the expense of problem objectives. A simple test of a constraining design process is to ask whether the process steps are significantly contributing to a better final design. If process steps are being accomplished for their own sake, they are a waste of the design team's time and the client's resources. Conversely, a design process which is too flexible and unstructured provides inadequate methodology for the design team to conceptualize solutions, compare alternatives, and finally choose a "preferred" system. Existence of a formal process can be a significant driver in keeping the design team on track and providing backbone to the seemingly unbounded world of systems design. Thus, the design process is a useful tool for the team in the management of complexity, which is inherent at some level in all design. The question therefore arises, what is the best design process for the problem at hand?

The design team was faced with a complex problem to be carried from the conceptualization stage of the lifecycle through to actual integration (and possible operation) of the system. A large portion of the system lifecycle development needed to be accomplished in a relatively short amount of time. The design team was required to make several iterations through the design process to move from initial conceptualization to detailed development. A flexible design process was used to handle this schedule-driven design problem.

The approach used by the team is outlined in Sage, who builds from the seven step process identified by Hall. Table 2-1 illustrates the correspondence of the Sage method to the Hall method.

Table 2-1: Problem-Solving Processes of Sage vs. Hall

Sage	Hall
Issue Formulation	Problem Definition Value System Design System Synthesis
Analysis	System Modeling System Analysis
Interpretation	Decision-Making Implementation/Documentation

The key to Sage's structured process is that Hall's seven steps are aggregated into three fundamental steps: *issue formulation*, *analysis*, and *interpretation*. These three steps define the overall system design process; each iteration through the system or subsystem level design incorporates these steps. The tasks within each fundamental step may be over- or under-emphasized as necessary depending on the problem or subproblem. Thus, the design team is not encumbered by implementation and documentation of a formal seven-step process for every problem or subproblem encountered. Sage's process accommodates the "time-to-market" approach which may require less emphasis on system synthesis and analysis for certain subproblems in favor of requirements satisfaction. It is important to note that although the process appears linear, there are feedback loops within every step and between steps. For example, during analysis it may be discovered that a significant objective was overlooked earlier, and this objective may then be incorporated into the value system design. Moreover, if requirements prove to be very difficult or costly to meet, they should be challenged and the problem redefined.

2.5.2.2 Issue Formulation. As a starting point for any design iteration, identification of problem characteristics and relevant issues must be accomplished. The

following information should be considered by the design team at this stage: actors involved the design process, groups affected by the issues or proposed solutions, fields of knowledge required to solve the problem, specific needs addressed by the problem, design alterables, constraints imposed, and cost and schedule considerations. The problem itself is isolated, quantified, and clarified. The system (or subsystem) to be developed is delineated from its surrounding environment. This abstraction of the environment consists of those elements which significantly interact or affect the system (or subsystem), but are beyond the design team's sphere of control (at this stage). Determination of what is the system and what is the environment allows identification and classification of important external interfaces. These tasks correspond to Hall's "Problem Definition" step. For the aircraft design problem, a design team has to determine what role the plane being designed will fill, or what need it will address. The corresponding step in the conventional conceptual aircraft design process is the "mission planning" phase.

Once needs are identified, development of system objectives begins. This process, Hall's "Value System Design", is the selection of a set of objectives that will guide the search for alternatives and be used for comparisons. It is the formalization of what is important to the customer. Value system design itself can vary in form. For some problems or subproblems, value system design may be the enumeration of specific measurables by which all alternatives will be judged. Thus, the determination of a preferred solution can be accomplished quantitatively. At a top-level systems architecting perspective, it is highly desirable to create an objective hierarchy with associated measurables to comprise the value system design; these measurables will be

weighted in the end to select a preferred alternative depending on the fidelity necessary to make sound decisions, an objective hierarchy may include only qualitative "values". These values represent those aspects of the design Depending on the fidelity necessary to make sound decisions, an objective hierarchy may include only qualitative "values". These values represent those aspects of the design. This objective hierarchy approach to value system design can be carried over to each problem or subproblem encountered as the design evolves and goes through repeated iterations of the design process. In some instances, a formal objective hierarchy may not be needed. In these cases, alternatives which are feasible (within constraints) may be chosen without searching for the preferred alternative. This satisficing approach may be desired for various reasons: tight schedule constraints prevent detailed alternatives comparisons, lack of reliable modeling data prevents accurate comparisons, or the utility of a preferred solution is comparable to that of other feasible solutions. For aircraft design, the design team has to decide on the weights to place on various measures of performance of the aircraft. Here is where the team has to initially decide what the trades between measures like payload, range, and cost should be.

The last phase of issue formulation corresponds to Hall's "System Synthesis" step. A set of alternative solutions is developed, through research, brainstorming, reverse engineering, heuristics, and other means. These alternatives should appear feasible, but need not fully comply with constraints at this stage (later investigation could reveal a feasible alternative was in fact infeasible; or conversely, a potentially infeasible solution may prove feasible). Determination of these alternatives is at the core of systems architecting. The actual development of alternatives in aircraft design can take many

forms. At this step, work can be divided among the aircraft design functional teams, with design managers responsible for integrating the components to generate an internally consistent alternative.

2.5.2.3 Analysis. Analysis includes the necessary system modeling and evaluation to make decisions regarding which alternatives to pursue further. System modeling is the development of means to evaluate performance of each alternative. Models are system abstractions used to evaluate the measurables for each objective. The systems evaluation phase is the use of modeling to quantify these measurables. At this stage, alternatives may be refined as necessary to improve performance.

System analysis may take place in many different forms. Construction of simulations, itemization of costs, development of prototypes, and engineering estimates are just some of the modeling methods available to the design team to quantify performance measurables. The goal of system analysis is to provide data for the decision making phase. Therefore, modeling is only necessary to the level of fidelity allowing differentiation of system alternatives.

The analysis of an aircraft design can take as many forms as the process by which it was created. Typically, the less in-depth the design is, the less analysis can be performed on it, and the less accurate the conceptual analyses will be.

2.5.2.4 Interpretation. Interpretation uses the information gained by analysis to make decisions and proceed to the next iteration of the design process. In the decision making phase, an alternative (or set of alternatives) is selected based on the analysis data and the value system identified earlier. There is an element of risk and uncertainty in the results obtained through analysis, and these uncertainties must be

considered by the decision maker. Dominated solutions should be identified and discarded from consideration. Some alternatives may be better in certain aspects, but less preferred in other areas. Decision making tools, utility theory, and objectives weighting are needed to settle on a preferred solution set. Interaction with the customer and chief decision maker is critical during this stage.

Once this set of alternatives is identified, planning for action is necessary. The design process to this point should be communicated and documented. Looking ahead to the next iteration, the allocation of resources and development of another design schedule is performed. The design process then begins another iteration, in which the problem is recast given the current solution set. If this is the final iteration, the final design is documented and implemented. A problem of the current method of conceptual aircraft design is that the decision to pursue a design or abandon it has to be made on a relatively small amount of lower quality data. Continued iterations of the conceptual design process will not necessarily increase the fidelity of the data. In order to get that sort of data, intermediate design is undertaken, necessitating a large increase in resources devoted to the project.

2.5.3 Other Problem-Solving Methods. One major advantage of Hall's problem-solving process is its independence from the lifecycle phase. The iterative process can be applied at each stage of the design. Some proposed systems engineering processes overlap the problem-solving and lifecycle phases to the point that differentiating between the two can be difficult, and the iterative nature of design is not as apparent. The "systems approach" identified by Eisner is a broad overview of the major steps necessary to develop a system, implicitly defining an overlapping problem-solving process and

design lifecycle. Eisner did not recommend this "systems approach" to be used as a problem-solving process in itself. In fact, the seven-step process of Hall is referenced. It was not used directly in the Aircraft Design Problem due to the single-dimensionality of the process. The single-dimensionality of this "systems approach" refers to the overlap of problem-solving steps and lifecycle phases. It should be noted that Eisner included feedback loops and iteration within his "systems approach" steps, although Eisner's work provided additional systems perspective to be used by the team. The following list categorizes the steps of Eisner's "systems approach":

Needs statement: Specify customer requirements.

Goals and objectives: Set the goals to achieve.

System requirements: Specify system requirements.

Specifications: Outline system specifications.

Synthesis of alternatives: Create and synthesize alternate plans.

Analysis of alternatives: Analysis alternate plans what if primary system does not work.

Formulation of evaluation criteria: Express how to test the system.

Update of specifications: Correct any changes made on specifications.

Building, testing, and acceptance of system: Build, test, and accept it if it fits the requirements.

Documentation and installation: Document everything made for next research and lifetime support.

Operation of system: Implement the system as in Hall's life cycle.

Modification and upgrade of system: Optimize the system by modification and upgrade to meet the demand in an effective manner.

Table 2-2: Problem-Solving Processes of Meredith, *et al.,* vs. Hall

Meredith, et al	Hall
Problem Definition	Problem Definition
Plan Approach	Value System Design
Allocate Resources	System Synthesis
Model and Analyze	System Modeling
Design and Evaluate Alternatives	System Analysis
Select Preferred Alternative	Decision Making
Implementation/Documentation	Planning For Action

2.5.4 Lifecycle Methodology. The use of an appropriate lifecycle model as part of the systems engineering process allows the design to be effectively managed as it progresses from a concept to actual implementation, and beyond. As with the formal problem-solving process, the use of a specified systems engineering lifecycle has advantages and disadvantages when compared to another lifecycle model. Selection of an appropriate lifecycle model at the outset of design is an important decision which guides the ensuing design process. Several lifecycle models were considered for use in the *Aircraft Design*, with eventual selection of a tailored model specific to this design.

2.5.4.1 Comparison of Various Lifecycle Models. This section describes the advantages and disadvantages of several lifecycle models considered for use during the *Aircraft* design.

2.5.4.1.1 Sage's Lifecycle Model. A relatively streamlined lifecycle model was proposed by Sage based on the three basic phases of design evolution. There exist feedback loop in this lifecycle model so that refinements can be made as the design evolves. The basic phases are the following:

- System definition.
- System design and development.
- System operation and maintenance.

The activities within each phase of the three-phase model are generally obvious, but may be explicitly listed for larger system designs. Sage proposed a 22-phase model based on the three-phase model to be used for large systems. This 22-phase model ensures certain objectives and design decisions are met before moving on to the next phase, thereby acting as both a system lifecycle and system management tool. Both the simplified three-phase model and expanded 22-phase model are shown in Figure 2-17. The advantage of Sage's simplified model is clear: it is easy to use and allows flexibility. The expanded model is more geared for large military/industrial projects and contains steps not applicable to this design. However, the simplified model had drawbacks with respect to the *Aircraft* Design. Aggregation of the majority of the design decisions into one "system design and development" phase made the natural course of design decisions and milestones less clear. This lack of explicit reference to the stages of design between

identification of need and system implementation precluded use of this model by the design team.

2.5.4.1.2 Hall's Lifecycle Model. Hall proposed a seven-phase system lifecycle which covered the entire system life, to include system phase-out. Figure 2-19 below shows how Hall's phases relate to those of Sage. The individual phases of Hall's model are described below :

Program planning. This phase results in formulation of activities and projects supportive of the overall system requirements. The system management plan is developed.

Project planning. Purpose is to configure a number of specific projects which together comprise the overall system program.

System development. This phase comprises the implementation of project plans through system design, resulting in preparation of architectures, specifications, diagrams, and other design material.

Production. This phase includes the activities necessary to physically realize the system.

Distribution. This phase results in the delivery of the system to the end user.

Operation. The ultimate goal of the system, this phase comprises use by the customer, to include maintenance and retrofit as necessary.

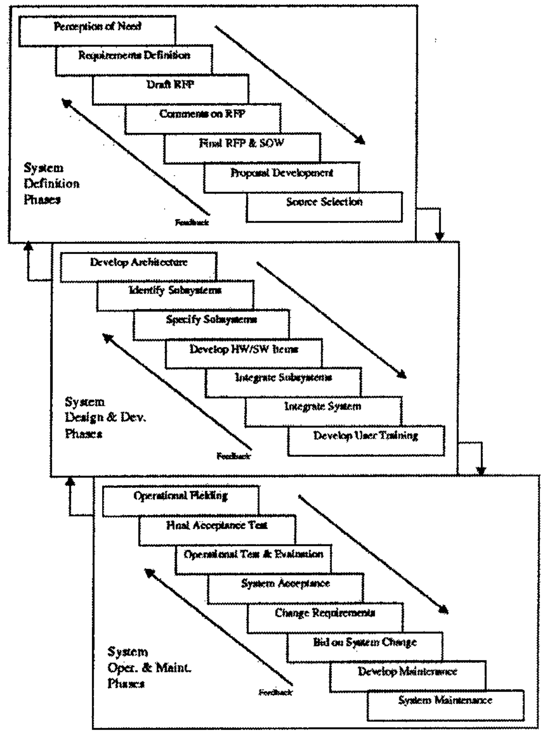


Figure 2-18: Sage's Lifecycle Model

Retirement. This phase, often overlooked in early planning, includes the phase-out and disposal of the system.

Although a comprehensive model, Hall's lifecycle was not used for this design for several reasons. Like Sage's three-phase model, the system development phase of Hall's model was not detailed enough to provide the design team with clear direction and objectives for this project. Furthermore, the system was designed and constructed in the same facility in which it would operate, making distribution an irrelevant step. As for the operations and retirement phases, they were not directly relevant to the design of this system, and were not addressed as separate lifecycle phases. As needed, continuation of the current design team's efforts through system retrofits and modifications was

accounted for. These upgrade programs would represent separate design problems and possibly separate thesis work, and thus were not included in the team's lifecycle model.

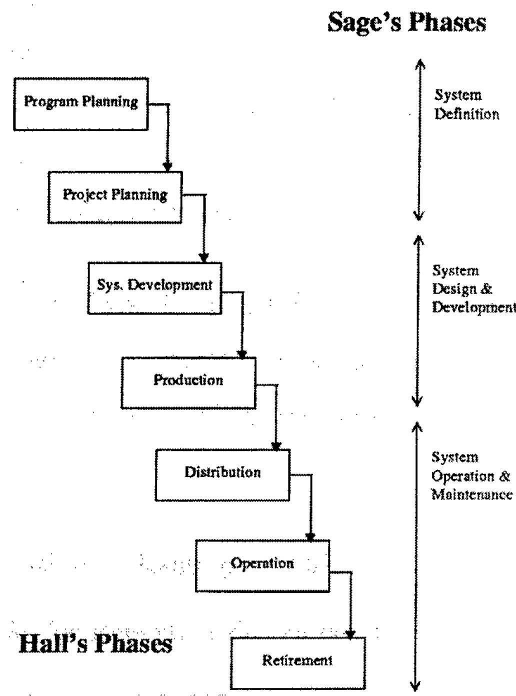


Figure 2-19: Lifecycle Models of Hall vs. Sage

2.5.4.1.3 Eisner's Lifecycle Model. Eisner presented a lifecycle model fairly similar to that of Hall. Despite more explicitly referencing concept exploration, this model still did not provide the design stage fidelity desired by the design team. Eisner's lifecycle consists of the following phases :

· Need development.

· Concept definition.

· Concept validation.

· Engineering development.

· Production.

· Operations.

2.5.4.1.4 DoD Lifecycle Model. The Department of Defense acquisition model described in the Defense Acquisition Deskbook was discounted for use in this design due to its relatively rigid review/milestone structure. As with the previous models discussed, the DoD lifecycle includes phases which were not relevant to the design team's academic charter. However, the DoD model provided useful guidance as to the delineation of design phases and use of design reviews.

2.5.4.2 Aircraft Design Problem Lifecycle Model. A system lifecycle tailored to the *Aircraft Design Problem* project was chosen to represent the design phases. The use of this model was driven by the following key factors used by the design team in their lifecycle modeling. The lifecycle should:

· Provide clear delineation of design progression.

· Allow natural breaks for important design decisions.

· Include only relevant lifecycle phases.

· Adequately accommodate a short design schedule.

The conceived lifecycle to meet these needs is shown in Figure 2-20. The following sections describe these lifecycle phases in detail.

2.5.4.2.1 Concept Exploration and Definition. Once a need has been identified and initial requirements have been defined, the system design process enters the first stage of the system lifecycle. This phase includes refinement of system

requirements, along with exploration of various concepts which can be designed to meet identified requirements. Emphasis is on top-level system architectures, with detailed design decisions avoided at this point. The focus of this lifecycle phase is to identify and differentiate broad solution classes. Through initial modeling, research, trade studies, and decision maker inputs, a class (or classes) of solutions may be identified which stands out from the rest. This solution class (or classes) can then be further refined and investigated during the next lifecycle phase.

2.5.4.2.2 Preliminary Design. In this lifecycle phase, the solution class(es) identified in Concept Exploration and Definition is (are) further refined. Subsystem level requirements are defined in this phase. Trade studies, research, and system modeling are used to determine which types of subsystems best meet the cost-effectiveness system goals. The output of this phase includes a system architecture complete with identified subsystem types, along with subsystem requirements and interface identification. In short, this phase translates system solution classes into subsystem solution classes, which are further defined and integrated in the next lifecycle phase.

2.5.4.2.3 Detailed Design. The subsystems are further designed in this phase. Detailed trade studies should be used to determine the exact subsystem architectures which make up the overall system. Integration and interface issues are resolved in this phase and the overall system is completely defined at this point, subject to change as system test and evaluation may require. The product of this lifecycle phase is a detailed functional system architecture with subsystems designed and integrated.

2.5.4.2.4 Final Design. The final product is described and documented for future users in this phase. Unresolved design issues are discussed. The design team makes recommendations and draws conclusions to aid future users and designers.

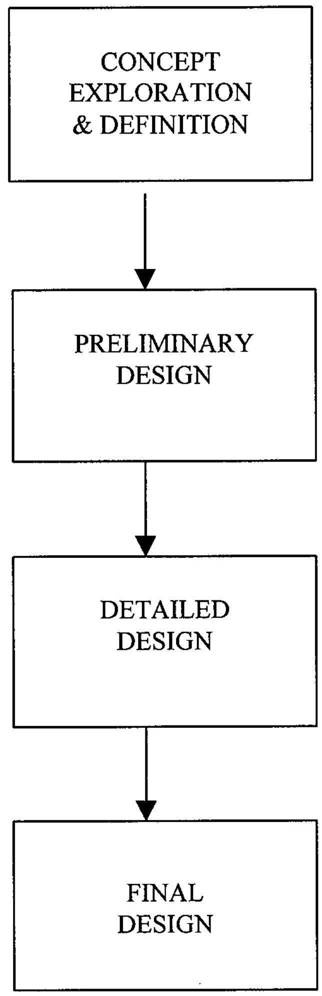


Figure 2-20: Aircraft Design Problem Lifecycle Model

2.5.4.3 Rechtin Perspective's on Systems Engineering Approach. A systems approach is one, which focuses on the system as a whole, particularly when making value judgements (what is required) and design decisions (what is feasible) (Rechtin, 1997:9). At a high design level, systems are the whole cooperatively working

with different kinds of elements, which produce outputs together that were unachievable by individuals. For example, only when elements are connected and working together do automobiles produce transportation, human organs produce life, and spacecraft produce information. These system-produced results derive almost solely from the interrelationships among the elements, a fact that largely determines the technical role and principal responsibilities of the system architect (Rechtin, 1997:10).

From an architectural point of view, system design and manufacturing have been a large field to produce the most competitive product on the market. Many changes are required to update the system as needed for its survival. Such required changes were largely a matter of continual, measurable, incremental improvement – a step at a time on a stable architectural base. The need was to make the classical manufacturing architecture more effective, that is, to evolve and engineer it.

Rechtin proposed a three-waterfall system lifecycle, which covered the entire system life, to include system phase-out.

Figure 2-21: Process Waterfall

Enterprise need and resources

Modeling

Engineering

Pilot Plan

Build

Certify

Production

Maintenance

Reconfiguration

Adaptation

Shutdown

Product Waterfall

Client need & resources

Conception & model building

Interface description

Engineering

Production

Certification

Operation & diagnosis

Evaluation & adaptation

Software Spiral

Functions

Form

Code

Test

Modern manufacturing can be portrayed as an ultraquality, dynamic feedback system intersecting with that of the product waterfall. The added tasks of the manufacturing systems architect, beyond those of all systems architects, include (1)

maintaining connections to the product waterfall and the software spiral necessary for coordinated developments, (2) assuring quality levels high enough to avoid manufacturing system collapse or oscillation, (3) determining and helping control the system parameters for stable and timely performance, and, last but not least, (4) looking farther into the future than do most product-line architects.

2.6 Unix Computer Software and Hardware Interoperability Problems

The aircraft model was produced by the Team using the Adaptive Modeling Language (AML). AML provides a great deal of model flexibility and is an asset to the conceptual aircraft design process. AML is supported and available to be hosted on many different software platforms which include PC computers as well as HP-UX, SGI/UNIX and SUN/UNIX platforms. When this effort began, AML training was performed using PC machines. The PC version of AML was first utilized to practice model building. The initial aircraft model was iteratively improved and a vast majority of modeling was accomplished using the PC version of AML.

However it soon became clear that since only the UNIX version of MSC.PATRAN was available, the UNIX version of AML must be employed in order meet the requirement of the thesis. Models previously coded in the PC version of AML are interoperable between PC and UNIX operating system platforms. This fact was soon confirmed by the Team, however, a vast majority of the thesis time and effort became focused on resolving problems performing AML functions within the UNIX platform (see Appendix B for more details on the UNIX AML process). For this reason, a return to the literature search is appropriate and consequently, this section is devoted to

gathering more information on UNIX software implementation difficulties and their solution.

2.6.1 Why the UNIX Support Problem Exists. The source of problems within many UNIX software versions may lie in in the dominance of PC software releases. The PC software version is usually the first released to the public. As such, it often receives the most attention with respect to quality. UNIX versions of software are subsequently released and tend to receive much less attention than their PC counterparts (DeLoach, 1997). With AML, this is not the case. The UNIX version of AML and many other AML special features are targeted primarily toward UNIX applications and UNIX AML predates the PC version of AML. MSC.PATRAN is also targeted primarily for UNIX, so the above argument does not seem to apply to MSC.PATRAN either.

There are also typically many more versions of UNIX software than PC versions which a given company may support. PC software is usually classified alone with different versions being required only for differing operating systems such as Windows95, Windows98 and Windows2000. However, UNIX systems are classified according to many categories such as SUN/UNIX (Sun Microsystems), HP-UX (Hewlett-Packard UNIX), Digital UNIX, or SGI/UNIX (Silicon Graphics, Incorporated). The various manufacturers have each constructed their own versions of Unix. Each category of UNIX has its own version of operating system which require its own form of support and must be addressed in subsequent software releases.

Each individual combination of a manufacturer's hardware plus a version of the manufacturer's operating system constitutes an architecture for that set of equipment. The number of architectures has increased exponentially with the increase in the number

of computers, software packages and users. Whenever a software package such as AML is created or updated, it must then be checked in many different architectures (Unix, 1996).

Today there is simply not time to check every possibility, therefore, if the new software package works for the major UNIX architectures, it is implemented across the board. Future problems reported from the other untested architectures are pursued as they appear (Unix, 1996). In other words, every architecture in the UNIX environment of today cannot be supported by every software company. So some UNIX architectures fall through the cracks and their software problems must be dealt with after the fact.

2.6.2 Solutions to the Unix Support Problem. The number of computers, users, operating system versions and software packages has also vastly increased in recent years to the point where the support workload is too large to be adequately served. For example, the thesis effort encountered three different AML software version releases in six months which operated on three different PC and UNIX platforms. The University of Waterloo Unix Support Advisory Group suggest that to re-attain excellence in support, an organization must focus on those tasks that will advance the overall state of the computer environment. They further suggest that tasks which detract from the ability to accomplish this must be minimized or eliminated. For example, software companies should support no more than two versions of any operating system and support only those computers that are running a supported version of an operating system.

Industry groups are working toward minimizing differences in UNIX versions by standardizing various interface layers. However, vendor conformance has been very difficult to achieve given the vendor proprietary environment which has evolved up to the

present time (Unix, 1996). Improvements in support of UNIX version software and customer service seem possible through means such as the standardized architecture initiative and should be pursued.

3. METHODOLOGY

3.1 Classical Approach

Systems engineering is the science of generating complex interactions of components which must all function together as a coherent whole. To meet this challenge, Hall proposed well-disciplined, recursive application of a standard, comprehensive, iterative, systematic design process to logically approach aircraft structural design. These steps are:

Problem Definition: The first task of the Systems Engineering Process (SEP) is to define or redefine the systems engineering problem through a requirements analysis. Requirements analysis is the essential subprocess which identifies top level system functional and performance requirements, utilization environments, and constraints which set the basis for total system development.

Value System Design: Value system design outlines the Chief Decision Maker values into a hierarchy of objectives, where objectives flown down from the top level in a well-structured manner. In the successive applications of the process these CDM objectives are revisited to account for changes at the system level that must flow down to lower levels. This set of objectives should drive all design efforts, and it must serve as the standard by which alternative solutions are evaluated. Since established objectives are in conflict, it is necessary to conduct objective trade studies and assessments. Trade studies are made on alternative sets of objectives/constraints. Recommendations are made and impact is described, based on assessed cost, schedule, performance, and risk for each alternative set. The objective baseline is validated with customer to ensure that

the baseline represents what the customer expects. Validation is accomplished by cross checking all customers to review requirements, and by reviewing appropriate enterprise, project, standard, and interface documentation. The validated requirement baseline is provided as inputs to functional analysis.

System Synthesis: The purpose of the synthesis is to translate the functional architecture into a physical architecture. The functions of the functional architecture are grouped and then allocated to physical system elements to form alternative physical solutions. Each alternative solution is evaluated by systems analysis to determine the recommended alternative and associated impacts on cost, schedule, performance and risk. Throughout synthesis, trade studies are performed for each alternative based on safety and environmental hazards, life cycle quality factors, technology requirements, standardization opportunities, off-the-shelf solution availability, make or buy, failure modes and effects and criticality, testing needs, and the design capacity to evolve, to provide future customer utility and product enhancement.

System Modeling: The only way to tell whether a component, subsystem, or the whole system can accomplish its intended functions and has the correct performance features is to test it. The cheaper and easier way to do it is to model the system and record and compare each phase output with your expectations and specs.

System Analysis: Examine the whole system and the decomposed subsystems according to the technical review and specifications. Identify the weakness in each state and recommend an alternative.

Decision-Making: Propose if the system is feasible on hand analysis to the Chief Decision-Maker. Tell him or her the advantages and disadvantages of establishing such a

system based on certain criteria. Recommend your solution alternative while highlighting various perspectives.

Implementation/Documentation: Document everything possible to help the system survival and provide improvement in subsequent studies..

3.2 Process Tailoring

3.2.1 Introduction. Any systems engineering process is a template. It is natural to tailor the systems engineering process to suit the needs of the designers. Sage's three-phase process, for instance, while practical to a wide variety of design problems of varying complexity and detail, is extremely generalized. Hall's methodology can be seen as placing specific objectives to be completed before continuing to the next step, which may or may not be applicable to a specific project.

3.2.2 Methodology. The 2000 Design Team felt that some thought must be applied to the systems engineering methodology employed. After reviewing Hall's and Sage's methodology, the Team decided to base its process on Hall's seven-step methodology. In previous years theses, the major complaints against Hall's seven step process were the lack of a step which narrows the design space towards a solution, and the relatively loose definitions of the steps themselves. The Team felt that since the process for generating a new aircraft design generally begins with a previous aircraft design, the systems engineering process would not have to converge towards a design – instead, this would be the province of optimization software. The steps could be defined by the team to minimize any potential for conflict or misunderstanding between the steps.

The Team felt that the first two steps of Hall's methodology were well defined and could be used "as written." The remaining steps needed tailoring to the Team's specific problem. Notably, for the first design iteration, AFRL/VA had only general ideas about what would constitute a good airplane design and a good airplane design system. The availability of tools for analysis and optimization were also not known, and would be influenced by aircraft model design. The Team (and the sponsor) did not have a good idea of what information would be present at the end of a design iteration. Therefore, at least initially, the design methodology would have to be written in such a way that the decision making and implementation processes would be decided by the Team and the sponsor after design analysis and optimization were performed.

The Team developed a six-step systems engineering process tailored for their design problem. An explanation of the steps in brief is below, and is followed by a more detailed elaboration in the succeeding sections.

3.2.3 Problem Definition. Any design work begins with the solicitation of the design problem the work will address. The problem definition for this effort can be taken from the thesis' original prospectus and the problem statement in Chapter 1.

3.2.4 Value System Design. Once the problem has been well-defined, the next step is to determine what criteria are important to the customer. The intent is to use insights into the customer's values to guide the design process and to evaluate and optimize the design. Hall is purposefully vague on how this can take place and what the end results of it would be. In a fully quantifiable environment, criteria can be divided into subcriteria until each subcriterion is measurable. Based on the range of scores expected in these subcriteria, weights can be assigned to each subcriterion, allowing

evaluated designs to be scored and ranked against each other. In an instance when the design team or customer is not sure about the relative importance of a suggested criterion, the solicitation of the customer's values still provides insight into the customer's understanding of the problem and guidance to the team. Since the design process is inherently iterative, the possibility exists that based on information gathered in future iterations, the value system design can change and become more quantifiable as design work goes on.

3.2.5 Alternatives Generation. Based on the information generated in the Alternatives Generation step, a model of the aircraft is designed in AML. AFRL/VASD wanted to ensure that the design presented to the suite of computer tools used for analysis and optimization would be feasible. For this reason, the Team was presented with a problem: how to ensure that the seed that starts the design process would be feasible. The team decided to use the output of the traditional conceptual design process as the first design to be analyzed by the software process. In future iterations, it is expected that the designs to be analyzed will be the output from the previous iteration, with slight changes.

Where Hall's System Synthesis step generates multiple alternatives, possibly with little regard to the feasibility of the designs, the Team's Alternatives Generation step creates one or a narrow number of alternatives which are feasible or close to feasible. The feasibility is demonstrated either by a relative of the design having been verified in the previous iteration or using the design as the output of the traditional conceptual design process.

3.2.6 Analysis and Optimization. After the model of the aircraft is coded into AML, the next step is to analyze the design to ensure that it will meet the set requirements and optimize the design so it will meet the requirements in an efficient manner. Analysis and optimization are different, but have been combined in this methodology since optimization is essentially smart, iterative analysis.

As currently designed, the methodology envisions three types of analysis/optimization. Volumetric analysis ensures that the AML model has enough interior room to load a particular payload. This analysis is done within AML, with the user creating geometric cargo objects after creating the geometric model of the aircraft, and looking for interference between the two. Cost analysis uses the information about the aircraft developed in the Alternatives Generation step to estimate its life cycle cost. Optimization is performed using ASTROS, a multidisciplinary optimizer. A finite element file containing connectivity information and materials properties of the region of interest of the AML model is generated and input into ASTROS. ASTROS can perform weight minimization of the structure given several different types of loading. Since optimization occurs one "discipline" at a time, ASTROS can develop several "optimal" designs from their one input design. ASTROS can change/generate the information used for the volumetric and cost analyses, so these analyses can be performed again if the results from ASTROS appear to drastically change the previous results.

3.2.7 Decision Making. After the aircraft design has been coded into AML, analyzed, and optimized, the end user is presented with the ability to judge the fitness of the optimized aircraft. This is done by applying the value system design developed previously to the output of the Analysis and Optimization step.

3.2.8 Implementation. Implementation is the last of the six steps. It involves reviewing the results of the iteration and making plans based on the results. If the output from an iteration is judged "good enough," the iterative design process can end. Otherwise, the information gathered in the iteration can be used change the design methodology and the class of alternatives examined in the next iteration. The Implementation step "wraps up" the iterative methodology.

3.3 Problem Definition

3.3.1 Problem statement. In order to meet the demand for more efficient aircraft designs such as the non-circular fuselage BWB design, designers must iteratively engage the aircraft design process in a multi-disciplinary environment. Any improvements which designers contribute in each cycle of the process are fed back into subsequent cycles.

The current aircraft design process is very cumbersome which vastly decreases its active contribution to iterative design improvement. Because the process is long and each step involves multiple discipline areas, communication among disciplines is extremely difficult. The initial aircraft design must be modeled to a high degree of definition, typically using a Finite Element Method software package. The FEM typically is very difficult to produce and extremely difficult to change. This is a great hurdle to streamlining and improving the aircraft design process. This process must be greatly accelerated to allow for consideration of a much larger aircraft design space if concepts such as the BWB are to become reality.

3.3.2 Problem Solution. Object-oriented computer software packages have appeared which greatly simplify the task of modeling and optimizing an aircraft design. Using such a package produced by TechnoSoft, Incorporated (TSI) called the Adaptive Modeling Language (AML) it is possible to rapidly generate a parametric model of a candidate aircraft design. The parametric model incorporates many aspects of the aircraft design including weight, cost, and planform. Parametric modeling allows many of the aircraft aspects to be defined in terms of a few primary dimensions. This subsequently simplifies experimental design changes in the candidate aircraft design. When a change in one dimension is made, it automatically causes design changes that ripple through the entire design. This encourages rapid improvement, evaluation, feedback and re-evaluation of many subsequent aircraft designs. Trade studies may then be generated based on a much greater information base.

The AML software is designed to recognize modeling objects that interface directly to a meshing software package known as PATRAN. PATRAN is then used to generate a geometric mesh of the design. Using the geometric mesh information, an input file, or deck, may be generated for use in the aeronautical structural evaluation/optimization software package known ASTROS. The Team seeks to construct such a model that will quickly progress from concept to evaluation with rapid feedback and flexibility of design change. This type of model is expected to demonstrate a marked improvement in the rapid design of efficient aircraft by reducing analysis effort and increasing the number of possible designs considered during the process.

3.4 Value System Design

Once the problem is sufficiently defined, the next step in the process is to design a value system in which the goals and priorities of the customer are defined. This ensures the customer requirements will be appropriately addressed. The thesis sponsor provided certain requirements that defined the value system for this effort. The desired aircraft model and its newly developed evaluation process must accomplish the following results:

1. Create a conceptual aircraft model within AML that is simple, flexible and easy to change (capable of rapid change from conventional Boeing 777 type design to a BWB design as well as intermediate designs).

2. The AML model must be capable of being geometrically meshed with quadrilateral elements (for finite element analysis) using the AML to MSC/PATRAN interface within AML.

3. Connectivity files containing node locations and element definition must be generated during meshing within AML.

4. Demonstrate conversion from the mesh connectivity files generated by AML to a form acceptable for finite element and other analysis using the ASTROS software.

5. Analyze the entire aircraft structure using ASTROS, including static loading deformation, element stress under loading, optimization of structural weight and member thickness, modal, flutter and aerodynamic analysis.

6. Change the aircraft design model from a conventional type aircraft to a BWB design (demonstrating model flexibility).

7. Regenerate connectivity files for subsequent ASTROS analysis run on the BWB design.

8. Analyze the aircraft design using ASTROS, including static loading deformation, element stress under loading, optimization of structural weight and member thickness, modal, flutter and aerodynamic analysis.

9. Demonstrate that the AML model reduces iteration time (Time required for model creation on second iteration must be less than time required for first iteration).

10. Establish, demonstrate and document the over-arching process to accomplish all the above items.

3.5 Alternatives Generation

3.5.1 Conventional Method. The Alternatives Generation step begins with the determination of a conventional aircraft design using the traditional conceptual design method explained in Chapter 2.

3.5.1.1 Introduction. Our design process begins with the mission requirement. The mission requirements are studied to identify the major requirements that drive the design. After identifying the mission requirements we will estimate initial aircraft sizing which includes the estimation of the total takeoff gross weight and the fuel weight. In this section, we will develop a conceptual sizing of such an aircraft with the process defined by Raymer.

3.5.1.2 Mission Requirement. The mission requirements are extremely important because they drive the design and are the yardstick by which the success of the design is measured. We will set the mission requirements and measures of merit at the beginning of the conceptual design process and compare the results at the end with them. As a mission definition, our purpose is to design an aircraft capable of flying 9,000nm with 800,000 – 1,000,000 lbs. total takeoff gross weight; we will assume a more efficient fuel consumption rate.

Our mission requirements include the range, total takeoff weight, payload weight, cruise speed and cruise flight level as shown in Table 3-1.

Purpose	: commercial transport aircraft
Range	: 9,000 nautical miles
Flight Level	: 36,000 ft.
Cruise Speed	: 0.85 Mach =547.7 knots
Total takeoff weight	: 0.8 – 1.0 M lbs.
Payload weight	: 180,000 lbs.

Table 3-1: BWB Design Mission Requirements

3.5.1.3 Initial Aircraft Sizing. Here many assumptions must be made in order to produce a feasible aircraft design. First we have to estimate total takeoff gross weight, which includes payload weight, fuel weight and empty weight. If we define W_0 as the total takeoff gross weight then the following equation summarizes the takeoff weight buildup.

$$W_0 = W_{payload} + W_{empty} + W_{fuel}$$

Where W_0 is the total takeoff gross weight, $W_{payload}$ is the passengers and cargo weights, W_{empty} is the empty weight of the aircraft which includes the structure, engines, landing gear, avionics, instruments, fixed equipments and anything else not considered a part of payload or fuel and W_{fuel} is the weight of the fuel required for performing the mission.

To simplify the calculation, both empty and fuel weights can be expressed as fractions of the total takeoff gross weight, i.e., (W_{empty} / W_0) and (W_{fuel} / W_0),

$$W_0 = W_{payload} + (W_{empty} / W_0)*W_0 + (W_{fuel} / W_0)*W_0$$

This can be written as follows,

$$W_0 - (W_{empty} / W_0)*W_0 - (W_{fuel} / W_0)*W_0 = W_{payload}$$

$$W_0 = \frac{W_{payload}}{1 - (W_{empty} / W_0) - (W_{fuel} / W_0)},$$

The only known is the payload weight since it is given in the mission requirements. The empty weight and fuel weights are the only unknowns. However empty weight and fuel weights are both dependent on total takeoff gross weight. Thus an iterative process must be used for aircraft sizing based on the equation above. Now the total takeoff gross weight, W_0, can be determined by estimating the empty weight fraction (W_{empty} / W_0) and the fuel weight fraction (W_{fuel} / W_0).

3.5.1.3.1 Empty Weight Estimation. The empty weight includes the structure, engines, landing gear, avionics, instruments, fixed equipments and anything else not considered a part of payload or fuel. The empty weight fraction, (W_{empty} / W_0), can be estimated by using historical data and trends. Figure 3-1 shows historical empty weight fraction trends for a flying boat, general aviation single and twin, sailplane powered and unpowered, homebuilt metal/wood and composite, agricultural aircraft, jet trainer, jet fighter, jet transport and military cargo/bomber. As seen from Figure 3-1 below, empty weight fractions change from 0.3 to 0.7 and diminish with increasing total aircraft weight. The type of aircraft also affects the empty weight fraction trends. For

example, flying boats have the highest empty-weight fractions because they need to carry extra weight for what amounts to a boat hull and military cargo aircraft have the lowest empty weight fraction. We also notice that different types of aircraft have different slopes of the trend lines of empty-weight fraction vs takeoff weight. Different aircraft develop different lines because different designs are dominated by specific considerations. The design constraints created by the desire for commercial transports to have low cost of operation with a high level of reliability are different from those imposed by jet fighters needing to be high performance, maintainable aircraft. Trend lines are all exponential equations based on takeoff gross weight. Since our BWB design is basically considered a subsonic cruise speed, we can use the (W_{empty} / W_0) ratio of 0.45 which is very close to Jet Transport and Military Cargo/bomber as a best estimated initial ratio.

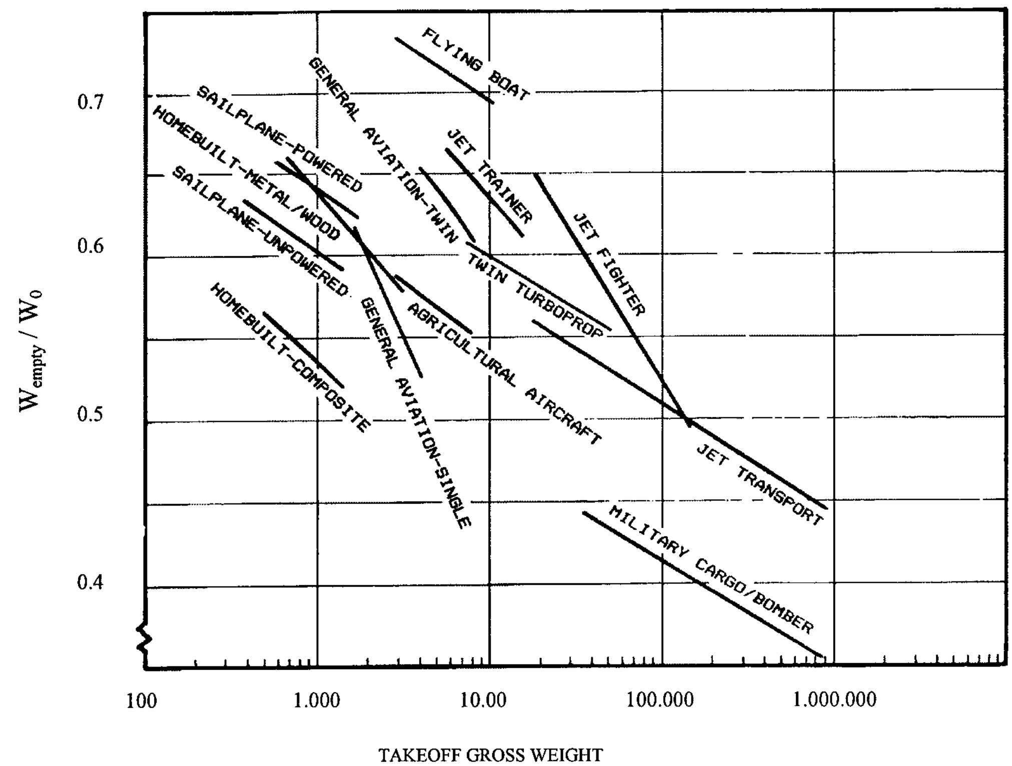


Figure 3-1: Empty Weight Fraction Trends (Raymer, 1989)

Empty weight fraction trend lines shown in Figure 3-1 are calculated by statistical curve-fit equations that presented in Table 3-2 below. These are all exponential equations based upon takeoff gross weight. C values in the curve-fit equations are small negative numbers, which indicates that the empty weight fractions decrease with increasing takeoff weight. The differences in exponents for different types of aircraft reflect the different slopes to the trend lines, and imply that some types of aircraft are more sensitive in sizing than others. A variable-sweep wing is heavier than a fixed wing, and is accounted for at this initial stage of design by multiplying the empty-weight fraction as determined from the equations in Table 3-2 by about 1.04. Similar fractions are also determined for the performance, efficiency, mission, length of takeoff run and

landing roll, materials used to build, volume of aircraft. Once deviation from the fraction occurs, we will trade off one of the feature above; poor performance, long takeoff run, inappropriate design for specific mission, etc. There will some modification for civilian types of aircraft design to build the military version and off course deviation from the fraction of that specific type in order to get the best result as performance, efficiency, etc.

For a conventional plane being built to our mission requirements, which would be very close to a military cargo aircraft or jet transport, we may choose a value of 1.01 for A and -0.06 for C in order to estimate (W_{empty} / W_0) in the statistical empty weight curve fit equation, $W_{empty} / W_0 = A\, W^C_0$.

$W_{empty} / W_0 = A W^C_0$	A	C
Sailplane-unpowered	0.86	- 0.05
Sailplane-powered	0.91	- 0.05
Homebuilt-metal/wood	1.19	- 0.09
Homebuilt-composite	0.99	- 0.09
General aviation – single engine	2.36	- 0.18
General aviation – twin engine	1.51	- 0.10
Agricultural aircraft	0.74	- 0.03
Twin turboprop	0.96	- 0.05
Flying boat	1.09	- 0.05
Jet trainer	1.59	- 0.10
Jet fighter	2.34	- 0.13
Military cargo / bomber	0.93	- 0.07
Jet transport	1.02	- 0.06

Table 3-2: Empty Weight Fraction vs W_0

3.5.1.3.2 Fuel Weight Estimation. The amount of fuel required to accomplish the given mission depends on basically three events. These are the mission to be flown, the aerodynamics of the aircraft and the engines' fuel consumption. After defining the mission for the aircraft, our preliminary estimate of the fuel weight can be found. For our particular aircraft, we consider the following four phases of its mission and will determine the fuel fraction for each phase shown in Figure 3-2.

Mission Profile:

Phase 1 – Warmup and takeoff

Phase 2 – Climb

Phase 3 – Cruise

Phase 4 – Land

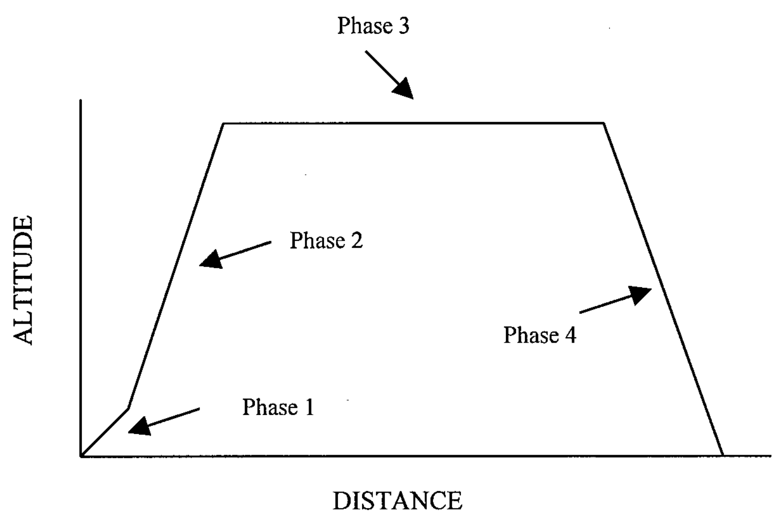


Figure 3-2: Mission Profile

During each mission phase, the aircraft loses weight by burning fuel. In order to estimate the required fuel fraction for initial sizing we need to calculate each mission phase weight fraction. For this particular cruise mission,

W_0 is the total takeoff gross weight of the aircraft at the beginning of the mission,

W_1 would be the weight at the end of the phase-1, which is warmup and takeoff, W_2 would be the aircraft weight at the end of the climb,

W_3 would be the weight after cruise and, finally

W_4 would be the weight at the end of the landing phase, which is also the end of the total mission.

Now we will determine the ratio of the final weight to the initial weight for each mission phase (W_i / W_{i-1}). If these weight fractions can be estimated for all of the mission phases, they can be multiplied together to find the ratio of the aircraft weight at the end of the total mission, W_x, divided by the initial takeoff gross weight W_0 . This ratio (W_x / W_0) can then be used to calculate the total fuel fraction required.

$$W_x / W_0 = (W_1 / W_0) * (W_2 / W_1) * (W_3 / W_2) * (W_4 / W_3)$$

We will use the historical average weight fraction values for the warmup, climb and landing phases, which are shown in Table 3-3 below for initial sizing.

	W_i / W_{i-1}
Phase 1 Warmup and takeoff	0.975
Phase 2 Climb	0.985
Phase 3 Cruise	TBD
Phase 4 Landing	0.995

Table 3-3: Historical Mission Segment Weight Fractions

We only need to calculate phase 3 cruise segment mission weight fraction by using the Brequet range equation: (Raymer, 1989)

$$R = (V*(L/D) /C)*\ln (W_i / W_{i-1}) \quad \text{or} \quad W_i / W_{i-1} = \exp (-(R*C)/(V*(L/D)))$$

where , R = range

C = specific fuel consumption

V = velocity

L/D = lift-to-drag ratio

In this equation we know the range and the velocity since they both are given in mission requirements. Then we only need to estimate specific fuel consumption, C, and lift-to-drag ratio, L/D.

3.5.1.3.2.1 Specific Fuel Consumption Estimation. Specific fuel consumption ('SFC' or 'C') is the rate of fuel consumption. Typical specific fuel consumption values for jet engines are shown in Table 3-4 below. We can use the SFC value of a high-bypass turbofan for rough initial sizing. On the other side since our BWB model will be designed with future advanced technology, we may use a lower fuel consumption rate of 0.4.

Typical Jet SFC's	Cruise
Pure turbojet	0.9
Low-bypass turbofan	0.8
High-bypass turbofan	0.5

Table 3-4: Specific Fuel Consumption

3.5.1.3.2.2 Lift-To-Drag Ratio Estimation. L/D is another measure of the aircraft design's overall aerodynamic efficiency. Since our BWB design model is a subsonic airplane, lift-to-drag ratio is most directly affected by two aspects of the design : wing span and wetted area. We need to estimate our L/D from the historical data for initial sizing.

In order to estimate L/D, we need to know the ratio of wetted area to wing reference area (S_{wet} / S_{ref}). Figure 3-3 shows different aircraft design approaches and the resulting wetted area ratios. As stated before, lift-to-drag ratio depends on the wing span

and the wetted area. We need to examine a new parameter, 'Wetted Aspect Ratio', which is defined as the wing span squared divided by the total aircraft wetted area. Figure-4 plots maximum L/D for a number of aircraft vs the wetted aspect ratio, and shows trend lines for some kind of aircrafts. By using Figure 3-3 for guidance, we can estimate the maximum lift-to-drag ratio from Figure 3-4.

For initial sizing, a wing aspect ratio of about 4 is selected. Comparing the examples of Figure 3-3, it would appear that the wetted area ratio (S_{wet} / S_{ref}) for our BWB design is about 2.5. This results in a wetted aspect ratio of 1.6 (i.e., 4/2.5). For a wetted aspect ratio of 1.6, Figure 3-4 shows that a maximum lift-to-drag ratio of about 20 would be expected. For cruise, a value of 0.866 times the maximum L/D, or about 18, is used for our initial aircraft sizing.

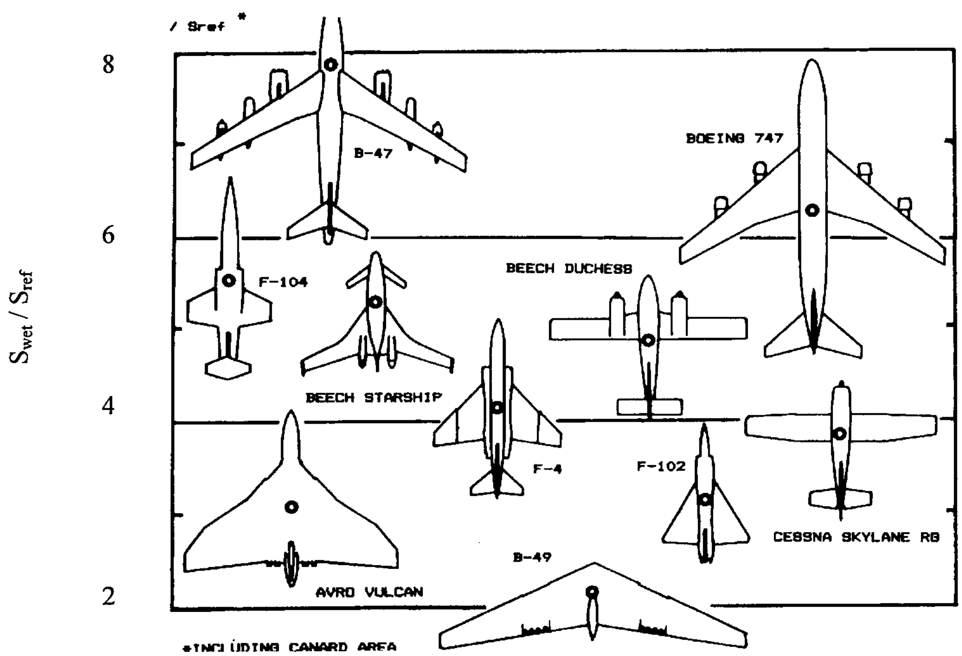


Figure 3-3: Wetted Area Ratios (Raymer, 1989)

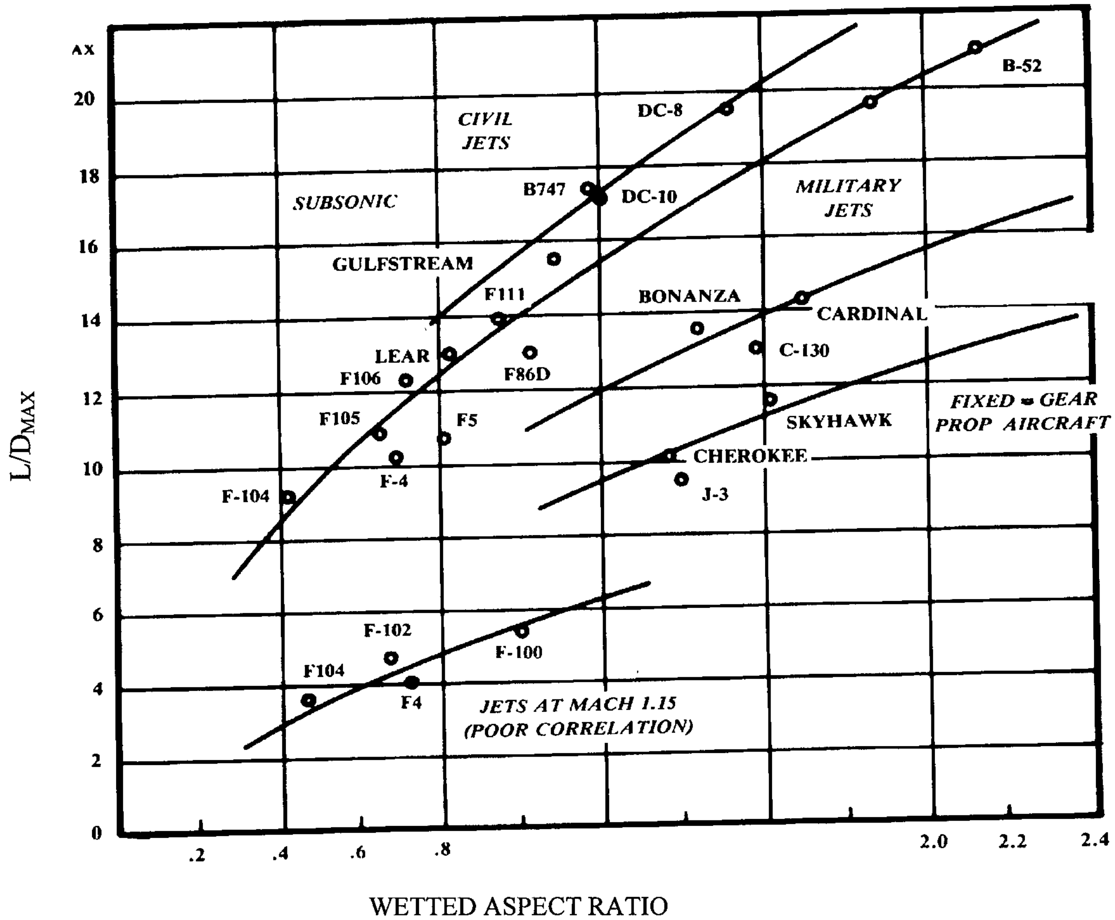


Figure 3-4: Maximum Lift-To-Drag Ratio Trends (Raymer, 1989)

3.5.1.3.2.3 Fuel Fraction Estimation. When we get the historical values and the equations for cruise phase, the mission phase weight fractions can now be estimated. As stated above, by multiplying them together, the total mission weight fraction W_x / W_0 can be calculated. The mission fuel fraction must be equal to $(1 - W_x / W_0)$. If we assume a 3% allowance for reserve and trapped fuel, the total fuel fraction can be estimated as,

$$W_{fuel} / W_0 = 1.03 * (1 - W_x / W_0)$$

3.5.1.3.3 BWB Initial Sizing Calculations. Using the statistical empty weight equation and the fuel weight fraction above, the takeoff gross weight can be found iteratively.

Mission Phase Weight Fractions

Phase 1. Warmup and takeoff $W_1 / W_0 = 0.975$

Phase 2. Climb $W_2 / W_1 = 0.985$

Phase 3. Cruise $W_3 / W_2 = 0.664$

where , R = 9.000nm = 54.684.000 ft.

C = 0.4 1/hr = 0.000111111 1/sn.

V = 0.85 Mach = 823 ft/sn = 561 mph. @ 36.000 ft.

L/D = 18

$$W_3 / W_2 = \exp\left(-(R*C)/(V*(L/D))\right)$$

$$= \exp\left(-(54.684.000*111.111\times10^{-6})/(823*18)\right) = 0.6635$$

Phase 4. Landing $W_4 / W_3 = 0.995$

Total Mission Weight Fraction is,

$$W_x / W_0 = (W_1 / W_0) * (W_2 / W_1) * (W_3 / W_2) * (W_4 / W_3)$$

$$= (0.975) * (0.985) * (0.6635) * (0.995)$$

$$= 0.6341$$

Fuel Weight Fraction is,

$$W_{fuel} / W_0 = 1.03 * (1 - W_x / W_0) = 1.03 * (1 - 0.6341) = 0.3769$$

Empty Weight Fraction is,

$$W_{empty} / W_0 = 1.01\ W_0^{-0.06}$$

$$W_0 = (W_{payload}) / (1 - (W_{empty} / W_0) - (W_{fuel} / W_0))$$

$$W_0 = (180.000) / (1 - (0.3769) - (W_{fuel} / W_0))$$

$$W_0 = (180.000) / (1 - (0.3769) - (1.01\ W_0^{-0.06}))$$

We need to do some iterations to find a reasonable W_0 in this equation. This is done by guessing the takeoff gross weight, calculating the statistical empty-weight fraction, and then calculating the takeoff gross weight. If the result doesn't match the guessed value, a value between the two is used as the next guess value.

W_0 Guess	W_{empty} / W_0	W_0 Calculated
950.000	0.4422	995.290
980.000	0.4414	990.775
985.000	0.4413	990.041
989.399	0.4412	989.399

Then, estimated weight fractions for initial sizing are,

W_0 = 989.399 lb.

$W_{payload}$ = 180.000 lb.

W_{empty} = 436.523 lb.

W_{fuel} = 372.905 lb.

3.5.1.4 Trade Studies. An important part of the design is the evaluation and refinement of the design requirements. The most critical design requirements are range and payload of the BWB design. So we need to check their effects on the total takeoff gross weight by recalculating the weight fractions using selected ranges and payload weights.

3.5.1.4.1 Range Trade. A range trade can be calculated to determine the effects on the total takeoff gross weight if the required range is changed. We will recalculate the weight fractions using 8000, 8500, 9500 and 10000n.m. and will size the aircraft separately for each of those ranges. Since the range only affects phase-3 cruise phase, other phase fractions remain the same. These calculations are shown below and the results are plotted in Figure 3-5.

<u>8000n.m. range:</u>

Phase 3. Cruise $W_3 / W_2 = 0.6945$

Total Mission Weight Fraction is,

$$W_x / W_0 = (0.975) * (0.985) * (0.695) * (0.995) = 0.6636$$

Fuel Weight Fraction is,

$$W_{fuel} / W_0 = 1.03 * (1 - W_x / W_0) = 1.03 * (1 - 0.6636) = 0.3465$$

W_0 Guess	W_{empty} / W_0	W_0 Calculated
850.000	0.4452	862.074
860.000	0.4449	860.786
860.697	0.4449	860.697

8500n.m. range:

Phase 3. Cruise $W_3 / W_2 = 0.6788$

Total Mission Weight Fraction is,

$$W_x / W_0 = (0.975) * (0.985) * (0.695) * (0.995) = 0.6487$$

Fuel Weight Fraction is,

$$W_{fuel} / W_0 = 1.03 * (1 - W_x / W_0) = 1.03 * (1 - 0.664) = 0.3619$$

W_0 Guess	W_{empty} / W_0	W_0 Calculated
900.000	0.4437	925.606
920.000	0.4431	922.832
922.491	0.4430	922.492

9500 n.m. range:

Phase 3. Cruise $W_3 / W_2 = 0.6486$

Total Mission Weight Fraction is,

$$W_x / W_0 = (0.975) * (0.985) * (0.695) * (0.995) = 0.6198$$

Fuel Weight Fraction is,

$$W_{fuel} / W_0 = 1.03 * (1 - W_x / W_0) = 1.03 * (1 - 0.664) = 0.3916$$

W_0 Guess	W_{empty} / W_0	W_0 Calculated
1.000.000	0.4409	1.074.643
1.060.000	0.4393	1.064.861
1.064.204	0.4392	1.064.204

<u>10000 n.m. range:</u>

Phase 3. Cruise $W_3 / W_2 = 0.6340$

Total Mission Weight Fraction is,

$$W_x / W_0 = (0.975) * (0.985) * (0.634) * (0.995) = 0.6058$$

Fuel Weight Fraction is,

$$W_{fuel} / W_0 = 1.03 * (1 - W_x / W_0) = 1.03 * (1 - 0.606) = 0.4060$$

W_0 Guess	W_{empty} / W_0	W_0 Calculated
1.000.000	0.4409	1.175.582
1.148.000	0.4372	1.148.313
1.148.268	0.4372	1.148.268

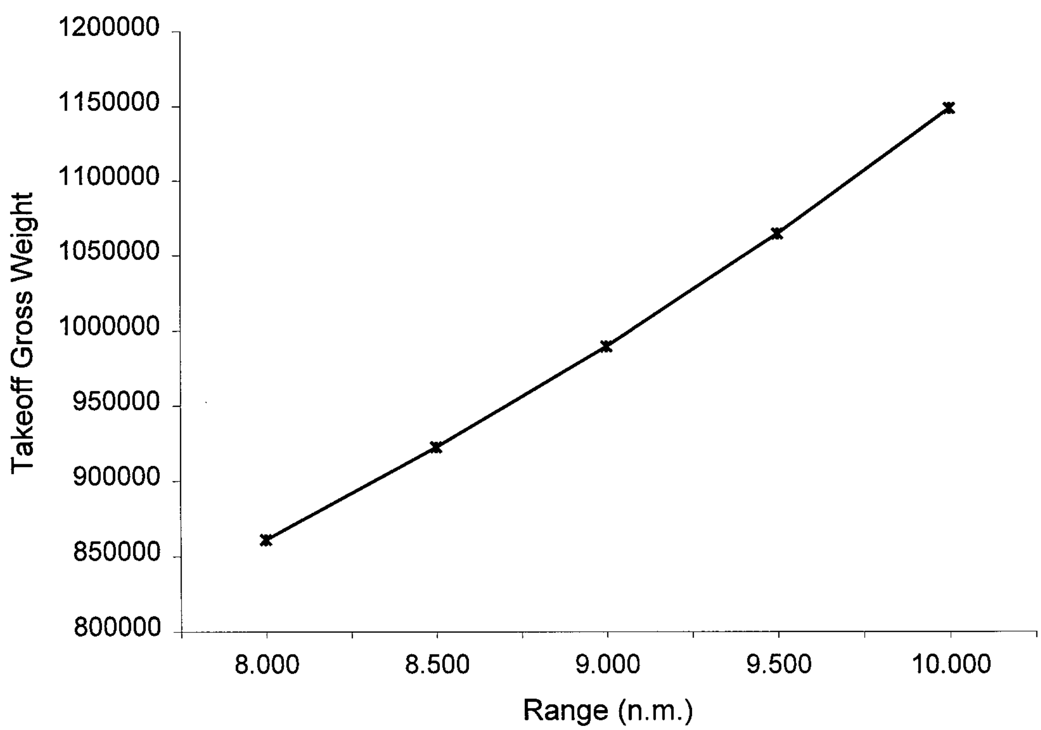


Figure 3-5: Range Trade

3.5.1.4.2 Payload Trade. In a similar fashion, a payload trade can be made. By assuming different payload weights we can calculate the total takeoff gross weight but the mission phase weight fraction and fuel fraction are unchanged. The calculations are shown below and the results are plotted on Figure 3-6.

Payload = 140,000 lbs.

$$W_0 = (W_{payload}) / (1 - (W_{empty} / W_0) - (W_{fuel} / W_0))$$

$$W_0 = (140.000) / (1 - (0.3769) - (1.01\ W_0^{-0.06}))$$

W_0 Guess	W_{empty} / W_0	W_0 Calculated
750.000	0.4486	802.095
790.000	0.4472	795.730
794.969	0.4470	794.970

Payload = 160,000 lbs.

$$W_0 = (W_{payload}) / (1 - (W_{empty} / W_0) - (W_{fuel} / W_0))$$

$$W_0 = (160.000) / (1 - (0.3769) - (1.01\ W_0^{-0.06}))$$

W_0 Guess	W_{empty} / W_0	W_0 Calculated

850.000	0.4452	899.387
890.000	0.4440	893.228
892.810	0.4439	892.810

Payload = 200,000 lbs.

$W_0 = (W_{payload}) / (1 - (W_{empty} / W_0) - (W_{fuel} / W_0))$

$W_0 = (200.000) / (1 - (0.3769) - (1.01\ W_0^{-0.06}))$

W_0 Guess	W_{empty} / W_0	W_0 Calculated
1.000.000	0.4409	1.097.580
1.080.000	0.4388	1.085.481
1.084.794	0.4387	1.084.794

Payload = 220,000 lbs.

$W_0 = (W_{payload}) / (1 - (W_{empty} / W_0) - (W_{fuel} / W_0))$

$W_0 = (220.000) / (1 - (0.3769) - (1.01\ W_0^{-0.06}))$

W_0 Guess	W_{empty} / W_0	W_0 Calculated
1.150.000	0.4372	1.183.428

1.175.000	0.4366	1.179.850
1.179.251	0.4366	1.179.251

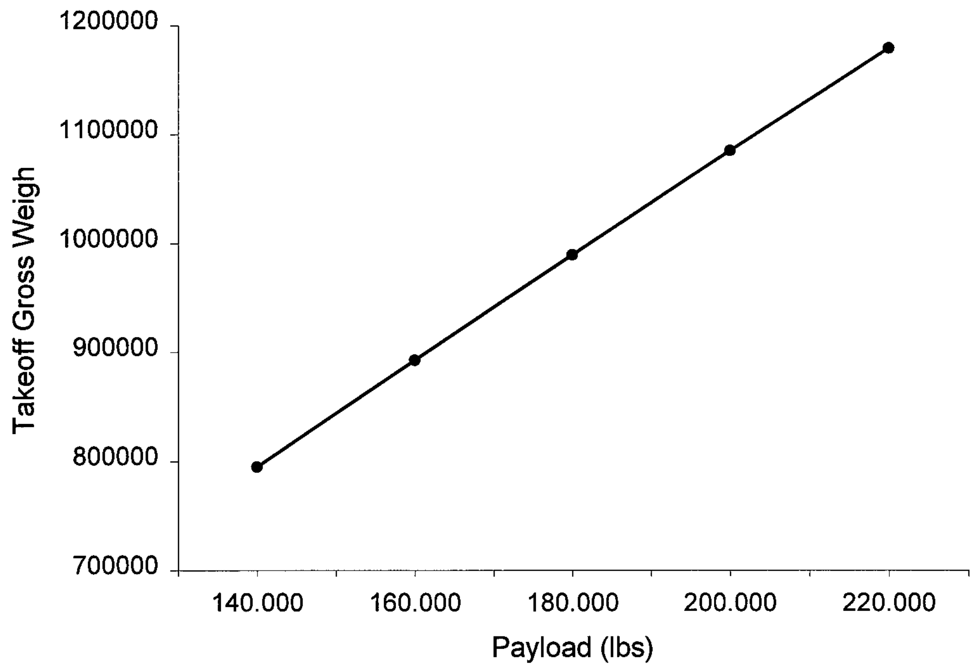


Figure 3-6: Payload Trades

3.5.2 AML Software Model Construction. The next step in the Alternatives Generation step is to develop a geometric model of the aircraft in AML. AML was specified by the thesis sponsors as the software vehicle for construction of the aircraft model. AML is an adaptive, object oriented, modeling language useful for knowledge-based concurrent engineering. It is a comprehensive modeling paradigm. AML is

produced by Technosoft, Inc., Cincinnati, OH and has been successfully used by many large companies. The Boeing Company, Lockheed Martin Aeronautics Company in Forth Worth, Texas and Lockheed Martin Missiles and Fire Control in Orlando, Florida have all used AML to successfully model aircraft and aircraft systems. The thesis sponsors have used AML for many years to model aircraft and it seems a natural choice to model aircraft for this conceptual effort.

AML supports only a part of the process that the Team seeks to demonstrate. The process begins with construction of a flexible, parametrically-defined, geometric AML model. Then using a new capability being developed by TSI, AML will interface with the geometry meshing software package known as MSC.PATRAN to generate a geometric model mesh complete with grid points and quadrilateral element connectivities. The model mesh data must then be collected from AML and formatted into an input data deck for finite element and optimization analysis using ASTROS. Therefore, to begin construction of the aircraft model, TSI's AML training was first completed. Much modeling practice was accomplished by following examples from the AML training manual.

AML is easy to use to develop simple models. Aircraft geometry and meshing geometry are topographically complex. AML facilitates complex geometric model development, but it required the Team to develop advanced skills. We document our progress from simple models to more complex geometric models in the following sections as a guide for others to follow.

3.5.2.1 AML Model Design Process Iteration 1: Simple Model. After brainstorming extensively with the project sponsor regarding the AML model, an initial

simple aircraft outline object or planform design was created. It consisted of eight points or stations labeled zero through seven (see Figure 3-7). These points and the lines connecting them defined a two-dimensional overhead view (planform) of one side of the aircraft. The points were initially given arbitrary numerical X-axis and Y-axis values to give the outline a reasonable passenger aircraft shape.

The outline object was coded by the Team, but the function of the outline object relied heavily on the outline object inheriting structure from a pre-existing AML object, the polygon-object. When the outline object was instantiated, it inherited the data structure and default properties of the polygon-object. Team written code modified the properties to create the desired object. Similarly, the polygon-object inherited its data structure and properties from other, more primitive objects. The ability to inherit the data structure and properties of another object makes creating user-defined objects easy and efficient. The evolution of the Team-coded model relied heavily on object-inheritance.

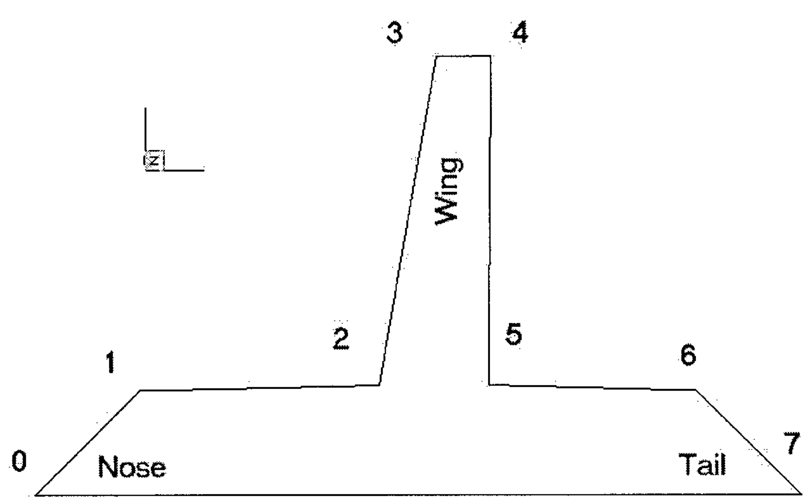


Figure 3-7: Aircraft Model Planform.

3.5.2.2 AML Model Design Process Iteration 2: Simple Model, Parametrically Defined. Next, the eight stations composing an aircraft outline were parametrically defined with respect to one another and with respect to certain defining distances. These defining distances may be changed at will, as the aircraft design is modified and the correspondingly defined dimensions of the aircraft planform will change accordingly. For example, changing the Fuselage-Width parameter causes a corresponding change in stations one through six (all the stations defined using the parameter Fuselage-Width). Table 3-5 lists the 24 defining parameters for a conventional aircraft in the last AML object iteration. All other dimensions are defined based on combinations of these parameters. The x and y locations of the eight stations are defined by functions which use the above parameters. Not all parameters influence the locations of each station; each station location typically depends on about six parameters.

Defining points based on functions of parameters introduced the concepts of demand driven calculation and dependency tracking into the Team's model. Demand driven calculation means that AML does not compute a calculated property until that property is demanded by the program. For instance, the station locations are not calculated until the user undertakes an action that requires them, such as drawing the planform, or inquiring what the value of the x term of station 6's location is. AML is able to determine what properties are dependent variables of other properties, so that if a property demanded by one object is dependent on the calculation of another object, this "upstream" object is automatically calculated when the "downstream" object or property is demanded. Because AML also keeps track of when the values or formulas in properties change, it only recalculates a property when its dependent variables change.

AML is able to limit its use of computer resources and the user gets a dynamic environment in which changes can be engaged interactively and rapidly.

Table 3-5: Parameters Used in Defining the AML Aircraft Model

Parameter	Property Value *(lengths in feet, angles in degrees)*
AIRCRAFT	
aircraft-length	177.0
aircraft-width	85.0
FUSELAGE	
fuselage-width	9.0
nose-angle	30
tail-angle	45
forward-fuselage-width-percent	0.95
trailing-edge-fuselage-width-percent	1.0
aft-fuselage-width-percent	1.0
station-0height-percent	1.0
station-1height-percent	1.0
station-2height-percent	1.0
station-5height-percent	1.0
station-6height-percent	1.0
station-7height-percent	1.0
WING	
sweep-angle	10
wing-taper	0.5
x-chord-length-at-root	24.0
leading-edge-location-percent	0.40
profile	"3125"
WING-BOX	
wing-box-chord-front-percent	0.20
wing-box-chord-back-percent	0.80
rib-quantity	6
spar-quantity	4
CARGO AREA	
fuselage-wall-thickness-factor	0.9

3.5.2.3 AML Model Iteration 3: Adding the Third Dimension. The next step toward an accurate aircraft model was to transform the two-dimensional, parametrically defined planform into a three-dimensional model. Using AML, a simple block shaped model was created by extruding the two dimensional model a given distance in the third dimension, or Z-axis. This model closely resembled a cookie cutter in the shape of the aircraft planform (see Figure 3-8).

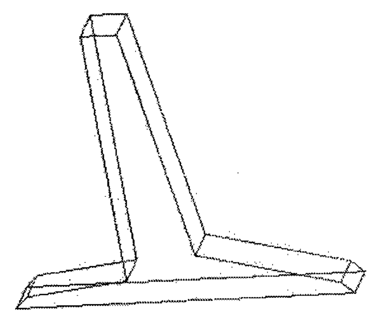


Figure 3-8: Extruded Three-Dimensional Aircraft Model

3.5.2.4 AML Model Iteration 4: Adding Three Dimensional Wings. AML possesses a native object which represents a National Advisory Committee for Aeronautics (NACA) four digit wing profile. Two parallel NACA wing profiles (one at the root of the wing and one at the tip of the wing) were defined based on the user supplied parameters X-Chord-Length-At-Root and Wing-Taper. The root of the wing is defined as passing through stations 2 and 5 of the fuselage (see Figure 3-9) and is thus attached to the fuselage of the aircraft. The complete wing was formed when the two NACA wing profiles were skinned using an instance of the skin-surface-from curves object. The aircraft wing was defined to have a root edge parallel to its chord edge for

model simplicity. A right and left wing were generated for the model as mirror images of each other.

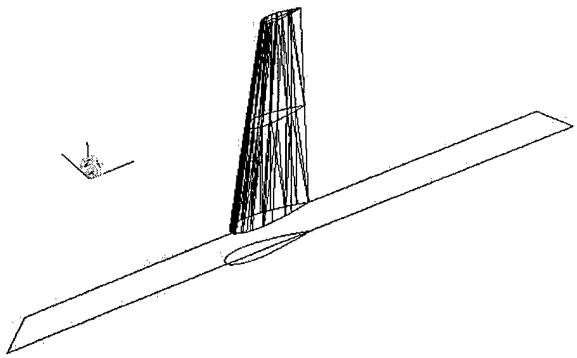


Figure 3-9: Aircraft Planform Plus NACA Skinned Wing

3.5.2.5 AML Model Iteration 5: Creating the Circular Fuselage. The aircraft model then required a fuselage. Six circles were defined which pass through stations 0, 1, 2, 5, 6 and 7 (See Figure 3-10). Circles at the nose and tail are too small to be seen in the picture.

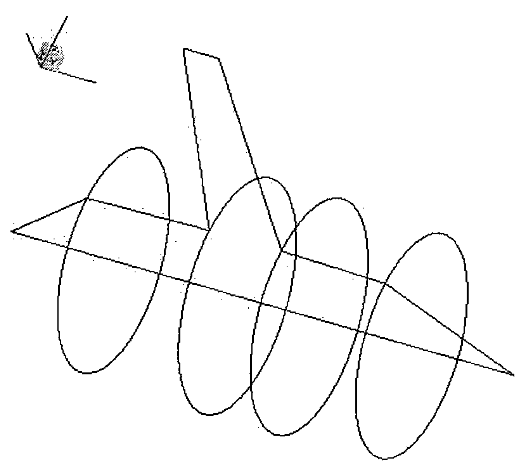


Figure 3-10: Planform and Circular Fuselage Cross-Sections

These six circles define the boundaries of the aircraft fuselage and are defined in terms of the aircraft parameters shown in Table 3-5. The circles' radii are simply the fuselage widths at the stations where the circles are placed. The object which creates the circles refers back to the planform object which calculated these values. The six circles required a skin, or covering, to become a fuselage model. Using another AML object class called skin-surface-from-curves an object was instantiated to act as the actual exterior, or skin, of the fuselage. The skin-surface-from-curves-object adds skin over the underlying circles and the resulting skin is gently curved in order to produce a smooth, continuous surface. To produce such a surface, the skin is made to curve out and beyond the radii of the underlying circles. This resulted in a "peanut" shaped aircraft fuselage. Figure 3-11 illustrates the skinned circular fuselage which is itself hollow.

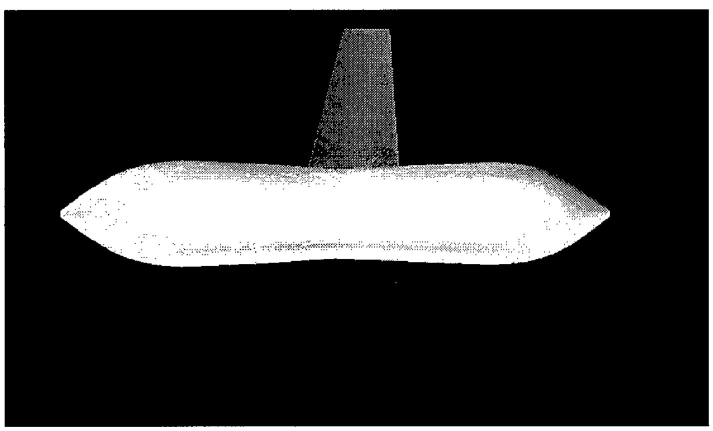


Figure 3-11: Skinned Circular Fuselage With Skinned Wings

3.5.2.6 AML Model Iteration 6: From Circular to Ellipsoidal Fuselage. In order to meet the Blended Wing Body requirement for the aircraft model, it became apparent that the fuselage model must be capable of stretching outward from a traditional, circular shape to more of an elongated ellipse until it finally became a blended fuselage-wing body shape. To model this capability, all of the six circle objects defining the

fuselage were changed to super-ellipse objects with user defined shape parameters that can be adjusted at will. Super-ellipse has parameters for the major and minor axes of an 2-D ellipse. If a third dimension is specified, the object creates a 3-D superellipse. Figures 3-12 and 3-13 illustrate the ellipses underlying the fuselage and the skinned ellipsoidal fuselage model.

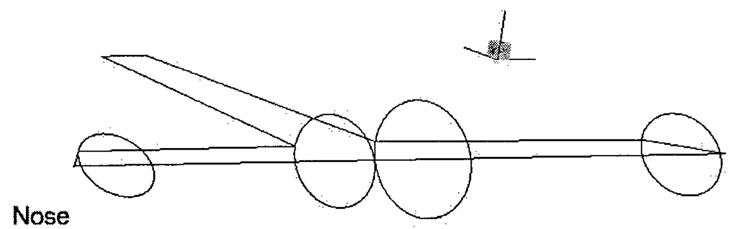


Figure 3-12: Fuselage Ellipses and Planform

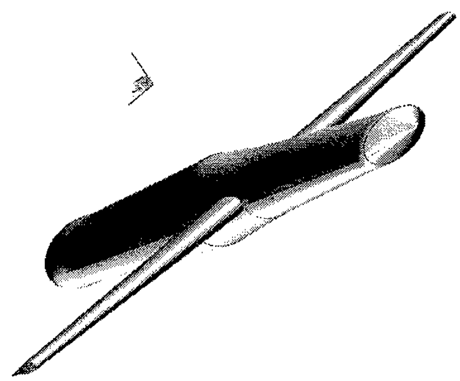


Figure 3-13: Skinned Noncircular Fuselage Model

3.5.2.7 AML Model Iteration 7: From Skinned Surface to Morphing Objects. The fuselage of the model when drawn appeared to be extremely peanut shaped. (See Figure 3-11.) This resulted from the use of the AML skin-surface-from-curves object. The object attempts to create a smooth, continuous surface between the user supplied curves. In the case of the previous model, AML draws the fuselage skin as a

smooth surface which passes through the six ellipses defining the fuselage itself. The object creates a smooth surface by keeping the radii of curvature extremely large. However, when only a small number of curves define a surface over a relatively large area, the skinned surface bulges out or compresses in between the defining curves . The result is a bulbous fuselage. Additional fuselage sections could be developed, increasing the number of points which the smooth curve would have to fit, and decreasing the peanuting tendency of skin-surface-from-curves. The Team had another option available to them.

To remedy the bulbous shape of the fuselage, the MDT-sponsored body-morphing object was used . Body-morphing works similarly to skin-surface-from-curves, but prior to attempting to skin a surface, it generates additional curves between the user supplied curves. Body morphing then skins a surface around the larger number of curves. Since there are fewer open areas between defining curves, the skin appears to be "tighter". Figure 3-14 illustrates the model shaping problem and its resolution.

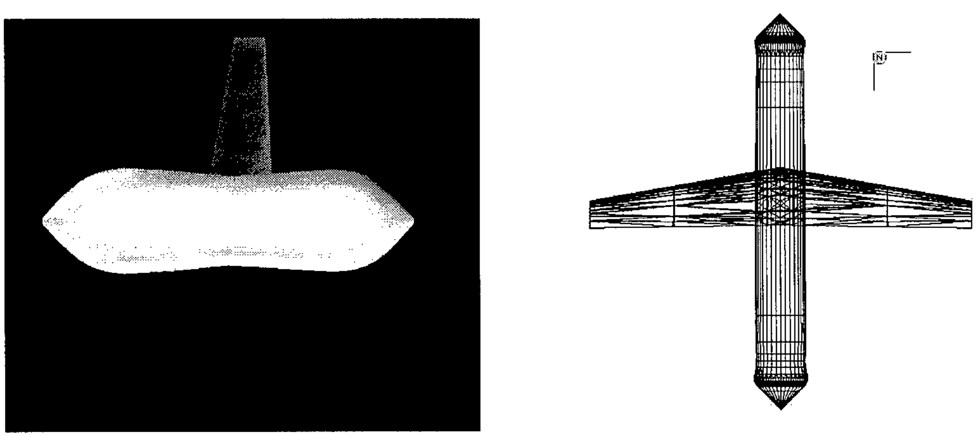


Figure 3-14: Fuselage Developed from Skin-Surface-From-Curves-Object (left) and Fuselage Developed from Body-Morphing-Object (right)

3.5.2.8 AML Model Iteration 8: Adding Cargo Objects to Aircraft Model. As part of the multi-disciplinary considerations given to the aircraft model, a payload-area object and a cargo object were created to model whether the aircraft would meet certain payload requirements. The cargo object was defined as a rectangular solid. The dimensions initially used are those of an M1 Abrams Main Battle Tank, thirty-three feet long by twelve feet wide by eight feet high. Figure 3-15 shows a single cargo object. The cargo objects can be instantiated as a series in various configurations. Figure 3-16 shows three cargo objects in line, as they might be loaded in a traditional fuselage aircraft. As the entire aircraft model is defined by the user, the cargo objects are compared to the payload area to determine the number of cargo objects the aircraft can hold.

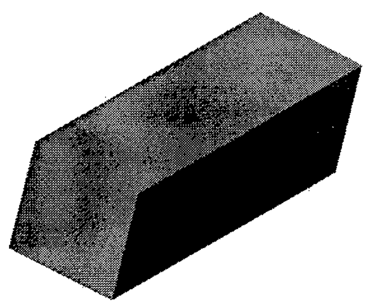


Figure 3-15: Single Cargo Object

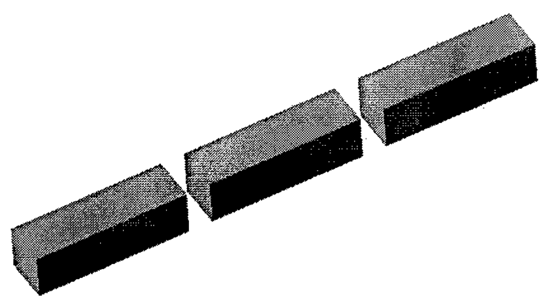


Figure 3-16: Cargo Objects In Line

3.5.2.9 Creating Hollow Wing Box. To create an aircraft model that ASTROS is able to analyze as a Finite Element Model, a sub-structure for supporting loads must be created within the previously generated aircraft model. After speaking with the thesis sponsors, it was decided that a wing-box object would be defined within the pre-existing wing. The wing-box acts as the load bearing member within the wing itself. The wing box formulation is an example of the use of Boolean objects in AML. The initial, hollow wing-box was defined within AML as the intersection of a rectangular

solid prism with the wing (see Figure 3-17). Later, additional ribs and spar objects will be added to the hollow wing-box to complete the structural wing support. AML's intersection-object creates a new object composed of the volume common to its constituent objects; since the constituent objects are solids, the intersection of them will be solid. (If the wing and wing-box prism had been defined as hollow surfaces, the common area of the two objects would be a collection of curves where the surfaces met each other.) The outside of the wing-box had been defined.

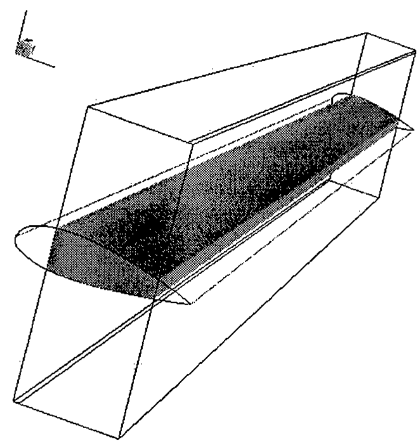


Figure 3-17: Definition of Wing Box as Intersection of Wing and Prism

3.5.2.10 Adding Ribs and Spars to Hollow Wing Box Structure. Once the outside of the wing box was defined, the internal structure of the wing box needed to be created. The number and location of the internal spars and ribs could have been defined in a number of ways. For instance, a maximum allowable distance between spars or ribs could have been defined, with the AML model placing the correct number of structures so that the maximums were not exceeded. Alternately, the locations of spars and ribs could be "hard coded" as locations from the root of the wing (for ribs) or the leading edge of the airfoil (for spars). The Team decided to allow the user of the model to specify the

quantity of spars and ribs wanted in the model. The AML model then spaces spars and ribs equidistantly. Figures 3-18 and 3-19 show the location and orientation of ribs and spars. Because the wing box exterior has already been defined, the box's four sides act as additional spars and ribs (see Figure 3-20).

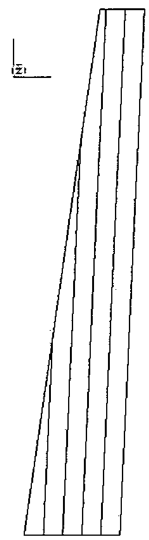


Figure 3-18: Spars and Wing Box

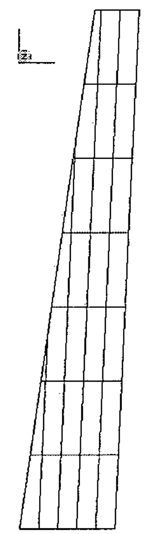


Figure 3-19: Spars, Ribs, and Wing Box

The spars and ribs were coded by creating large solid 2-D sheet-objects with the location and orientation of the desired ribs and spars. Then, each sheet was individually intersected with the (solid) wing box. The resulting intersection was the portion of the sheet within the wing box, or the spar or rib. The individual spars and ribs were grouped together with the shell of the wing box via a Boolean union-object, which creates a single geometry out of all areas of its constituent objects. This model now resembled an actual wing box sub-structure of an aircraft. The AML geometry was then ready to be tagged and meshed using the AML to MSC.PATRAN interface so that the AML results could be fed into ASTROS for structural optimization.

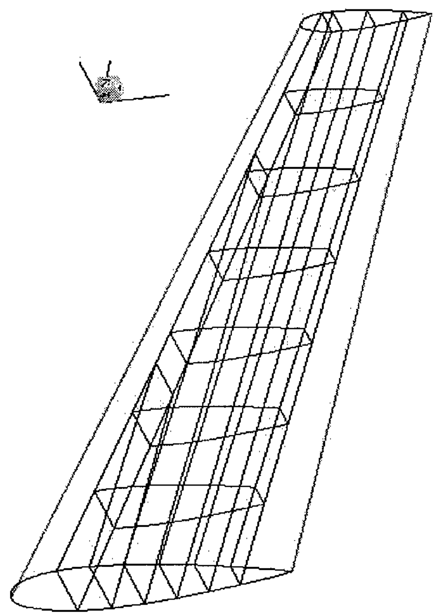


Figure 3-20: Wing Box with Ribs and Spars Compared to Wing

3.5.2.11 Tagging and Meshing the AML Model. This AML model also inherits from tagged-object. Tagged-object identifies what portions of the model are to be meshed and what mesh characteristics apply to each portion. Parameters for the tagged object control the fineness of the mesh and what information is saved about the mesh, for example, the 2-D connectivity of the mesh. Most geometric and Boolean objects in AML have a tagged- counterpart, which allow them to be meshed. The AML code was rewritten so that the wing box and its internal ribs and spars were tagged objects. To ensure continuity across the mesh, the entire wing box structure was incorporated as a single union-object and was meshed at once. The model was meshed first using AML's native triangular element meshing capability. The model was then quadrilaterally meshed using the AML-PATRAN interface. More detailed instructions on how to generate a mesh and the resultant connectivity files are found in Appendix B.

To gather mesh data, a query-object must be created which gathers the relevant mesh information about a particular object. Since the Team was interested in getting information from individual components of the wing box (ribs and spars were to be modeled as different types of FEM elements than the outer skins), individual query-objects were established for each rib, spar, and exterior surface. Consistent with AML's demand-driven calculation architecture, files detailing the connectivity of each component were created only when the query-objects were inspected or drawn. The results (in terms of element grid locations and connectivity) were saved to an output geometry file. The meshed model automatically adapts itself to any parameter selection as the configuration is transformed from a conventional design towards a BWB. These AML results completed one iteration of conceptual aircraft design within AML. The results were then formatted into an ASTROS input deck file for structural analysis of the conventional design wing box. The ASTROS analysis process is described in detail in section 3.6.2.

3.5.2.12 Process Iteration. The AML process was repeated at this point. The AML model was rapidly changed from a conventional design to a BWB design simply by adjusting the appropriate parametrically defined aircraft dimensions in AML and then repeating the tagging and meshing step found in section 3.5.2.11. Figure 3-21 below illustrates the BWB design which was generated from the conventional wing design within AML in less than one hour.

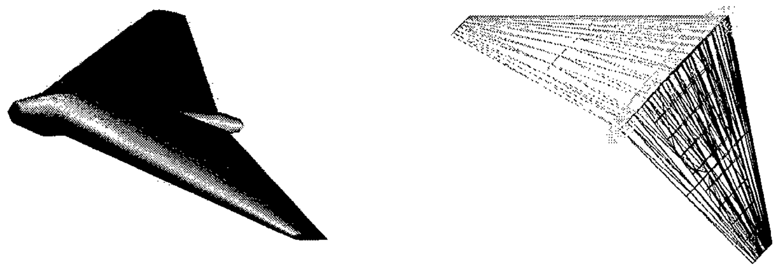


Figure 3-21: BWB Aircraft Design (left) and Wing Structure (right)

Table 3-6 contains a listing of the parameter values used to define the modified BWB design. Only 9 of the 20 parameters controlling the size of the aircraft were changed. Additional spars and ribs were added due to the increased size of the wing.

Table 3-6: Parameters Used In Defining Modified, BWB Model

Parameter	Value *(lengths in feet, angles in degrees)*
AIRCRAFT	
aircraft-length	85.0
aircraft-width	85.0
FUSELAGE	
fuselage-width	9.0
nose-angle	30
tail-angle	35
forward-fuselage-width-percent	0.50
trailing-edge-fuselage-width-percent	0.50
aft-fuselage-width-percent	0.3
station-0height-percent	1.0
station-1height-percent	1.0
station-2height-percent	1.0
station-5height-percent	1.0
station-6height-percent	1.0
station-7height-percent	1.0
WING	
sweep-angle	40
wing-taper	0.2
x-chord-length-at-root	65.0
leading-edge-location-percent	0.15
profile	"3125"
WING-BOX	
wing-box-chord-front-percent	0.20
wing-box-chord-back-percent	0.80
rib-quantity	8
spar-quantity	6
CARGO AREA	
fuselage-wall-thickness-factor	0.9

The AML mesh results completed the second iteration of conceptual aircraft design within AML. Modeling the second–iteration, modified BWB design in AML

required far less time and effort than the initial model constructed from the ground up. Changing the value of a few of the parametrically defined aircraft dimensions allowed the conventional aircraft model constructed in iteration one to be transformed to a reasonable BWB aircraft design in less than an hour. This compares quite favorably with the painstaking two months required to assemble the AML objects and develop the initial conventional aircraft design in iteration one. Figure 3-21 above illustrates how the AML aircraft model changed when 9 of the 24 parameters were changed. The figure shows the BWB aircraft model, structural wing design and wing-box design. The mesh data was again formatted into an input file for analysis in ASTROS and is described in the next section.

3.6 Analysis and Optimization

3.6.1 Volumetric Analysis. Volumetric analysis is a simple form of analysis performed on the AML-created aircraft model. It means checking that the aircraft, as designed, is suitably sized to carry the required cargo. Volumetric analysis compares the fuselage volume against the volume of the required cargo object(s). If interference is noted (if a cargo object "pokes" out of the fuselage's cargo area, for instance), then the design fails to meet one of its requirements, and should be revised.

The Team developed several geometric objects in AML to assist the analysis. As with all of the AML code developed, the major parameters can be changed, either in the code directly, or interactively during program run-time. The cargo-hold-object replicates the portion of the fuselage available for carrying cargo. The diameter of the cross-section

is determined at several stations along the plane and is a percentage of the outer diameter of the plane. The front and back of the cargo-hold-object are the stations where the nose and tail sections of the plane begin. Figure 3-23 shows a comparison of the fuselage to the cargo hold.

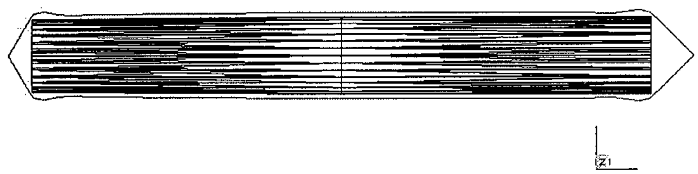


Figure 3-22: Cargo Hold (interior faceting) vs. Fuselage (outer line)

The cargo-object contains several subobjects itself – one is a single instance of the cargo object of interest and others are various configurations of multiple cargo objects, aligned in various orientations (see Figure 3-24). The user of the software needs only to instantiate the cargo-hold-object and one of the configurations of the cargo objects to see if the aircraft model meets the volumetric requirement. Appendix B includes information on the procedure used within AML to perform this check; Appendix C includes the AML code and comments.

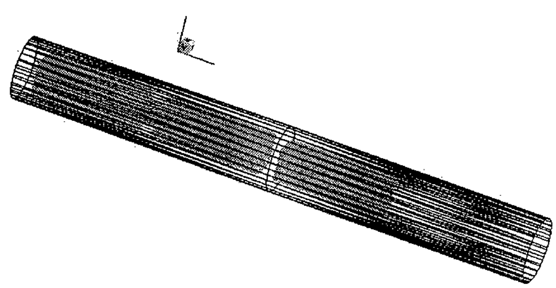


Figure 3-23: Cargo Hold With Cargo Objects

3.6.2 ASTROS Optimization. ASTROS is a finite element analysis tool capable of performing many different analyses including static, aeroelastic, flutter, modal and optimization of aircraft structural thickness and weight. ASTROS is the end evaluation tool for the conceptual aircraft design developed by this thesis effort. It was recommended by the thesis sponsor as the state-of-the-art in aeronautical design analysis tools. The Team used ASTROS to perform analysis on the AML created model that had first been meshed for finite element analysis using the AML to MSC.PATRAN interface. Below are the steps followed by the Team to perform finite element analyses of the AML aircraft model using ASTROS.

3.6.2.1 First Iteration: Forming the ASTROS Input Deck File for Conventional Wing-box Structure. After being meshed using the AML to MSC.PATRAN interface, the conventional aircraft wing-box model was ready for analysis. The structural analysis of the aircraft wing-box design occurred using ASTROS as specified by the sponsor. Additional details and background regarding ASTROS may be found in Appendix A. The meshing data generated in AML was composed of node locations and quadrilateral element connectivities defined in terms of the nodes. The mesh data was formatted into an appropriate ASTROS input deck, or text file, with a ".d" suffix.

QUAD4 and SHEAR elements were used in the finite element model. The QUAD4 element is a membrane-bending quadrilateral element and the SHEAR is a two-dimensional quadrilateral element that resists only in-plane shear forces. The entire upper and lower surface of the wing-box were assigned to be QUAD4 elements and the elements on the remaining ribs and spars were assigned to be SHEAR elements. A small

number of triangular membrane-bending elements (TRIA3) were generated in places where the quadrilateral elements would not fit (see Figure 3-25). These assignments were made following the recommendations of the sponsor. See Appendix A for additional details on ASTROS and the ASTROS input file format.

A material property for the model was assigned as aluminum and coded into the input deck. Although the team estimates that composites will likely be used in structural members in an actual aircraft design, their non-isotropic material properties make their representation in ASTROS difficult and beyond the scope of this effort. The coding of aluminum members will show the efficacy of the tools used and the Team-designed process. An equivalent three G wind drag loading (three times the aircraft takeoff weight) was applied to the wing through an element at the tip. A two and a half G loading due to fuselage weight on the wing was also applied as an upward (positive Z-axis direction) force applied at the wing tip (see Figure 3-25). A two and a half g-force loading is equivalent to a force three times the gross take off weight of the aircraft acting over the entire aircraft. Therefore to model this loading on one wing, half the gross take off weight of the aircraft was first modeled as a distributed load acting over the entire wing span. For simplicity, this distributed loading was divided by the length of the aircraft wing (measured from the centerline of the aircraft where structurally joined to the bulwark). The resulting point loading (of half the gross take off weight of the aircraft divided by the wing length) was applied at the wing tip and modeled as a wind drag force. The downward loading due to the force of the fuselage on the wing was modeled as half the three G wind drag force calculated above.

Gross Take Off Weight of Aircraft = 989,000 lbs.

2.5 G loading = 2.5 * GTOW = 2,473,500 lbs.

Half of 2.5 G loading = Force on one wing = 0.5 * 2,473,500 lbs. = 1,236,750 lbs.

Fuselage force on one wing divided by 16 application nodes = 1,236,750 lbs./16 application nodes = 77,300 lbs. applied per node.

Drag force on one wing is 1,236,750lbs/286 application nodes = 44,170 lbs./node.

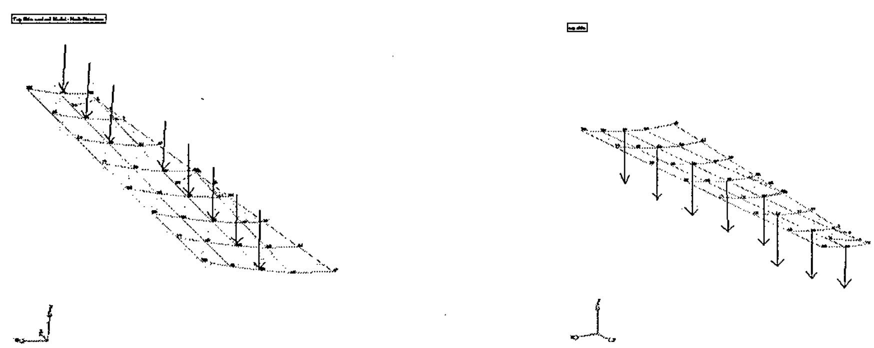


Figure 3-24 Top Skin Wing Loading (left) and Bottom Skin Wing Loading (right)

Single point constraints for all six degrees of freedom were applied along the root rib of the wing-box to simulate fixing the wing-box at the root nearest the fuselage. ASTROS was then utilized to analyze displacements of the wing elements due to the static loading and to optimize the thickness of the wing-box support structure members. The ASTROS output described displacements of each finite element due to static loading applied to the wing-box structure. The ASTROS optimization run optimized the wing design for thickness of all structural support members (all the lattice work planes that compose the wing-box). ASTROS optimizes the wing structure relative to sets of stress constraints imposed at each finite element. The sponsor provided useful advice on how to choose and apply the stress constraints in order to obtain a feasible optimal solution. The sponsor suggested distributing the drag force and force of the fuselage at various point along the wing. The ASTROS analysis results may be observed in Appendix C

under the ASTROS file section. The ASTROS results files are all the files with ".prt" suffixes.

3.6.2.2 The Second Iteration: Forming the ASTROS Input Deck for BWB Design Wing Structure. After being meshed using the AML to MSC.PATRAN interface, the modified BWB design wing structure was ready for analysis. The mesh data had to again be formatted into an ASTROS input file and the structure analyzed. The loading displacement and stresses of all elements were obtained under static analysis. The optimization analysis minimized the required weight of the wing structure.

The results of the ASTROS optimization for the conventional and the BWB wing structures were given to the sponsor for comparison and analysis. The ASTROS optimization routine was constrained by a maximum allowable stress which was user specified. Therefore an infeasible wing structure weight is one that does not meet the stress constraints given the applied loading. The conventional wing structure first optimized by ASTROS began at a supposed, infeasible weight of 5,800 lbs. and the ASTROS optimization routine converged in 11 iterations to an optimal (minimum) wing structural weight of 10,700 lbs. The BWB wing structure was then optimized using ASTROS. The BWB wing (aircraft) structure began at a supposed, infeasible weight of 18,500 lbs. and the ASTROS optimization routine converged in 11 iterations to an optimal (minimum) wing structural weight of 28,400 lbs. The complete ASTROS results are included in Appendix C under the ASTROS codes section. All ASTROS results files have ".prt" suffixes. The time of the entire second iteration of BWB design analysis was recorded (including AML model time and ASTROS analysis time) and compared to the time to complete the first iteration with the conventional aircraft and wing-box design.

The second iteration took dramatically less time at 1 day compared to 3 months for the first iteration. The recommendation and conclusion section in Chapter Five contains advice on how to iteratively continue this effort .

3.6.3 Life Cycle Cost Modeling. The LCC model for aircraft cost was estimated using parametric equations. Such equations are based on the cost of previous aircraft and are applicable to preliminary cost estimates which occur before many details are known about a conceptual aircraft design. The flyaway cost of an aircraft is defined as the total production cost including the cost for the airframe, avionics and engines. The LCC of an aircraft includes aircraft airframe, avionics, engines, O&S, and disposal costs. Because its cost is considered insignificant compared to the LCC cost, the Team ignored the disposal component of LCC. For the LCC estimate, an acquisition of 100 aircraft was assumed. This is slightly less than the current number of each type of transport aircraft in the USAF inventory. Using the best examples of the various component cost estimating relationships found below, the Team compiled a LCC estimate for the conceptual BWB aircraft conceived in the Alternatives Generation section above (section 3.5).

3.6.3.1 Raymer General Aircraft Flyaway Cost Estimate. The Defense Contractors Planning Report (DCPR) weight of an aircraft airframe equals approximately 70% of empty aircraft weight. \$150-\$300 per pound of DPCA weight of aircraft yields a rough flyaway cost estimate:

$$DPCR = 0.7 * W_o = 0.7 * 436{,}500lbs = 305{,}550lbs$$
$$COST_{FY88} = 305{,}550lbs * \$300 / lb. = \$91M$$
$$COST_{FY99} = COST_{FY88} * (1.645 / 1.15842) = \$91M * (1.4200) = \$129.2M$$
$$COST_{FY99corrected} = COSTFY99 * TECH = \$129.2M * 1.75 = \$226.1M$$

where W_o is the empty weight of the aircraft calculated in section 3.5 above, $COST_{FY88}$ is defined as the unit flyaway cost per aircraft in FY 88 dollars, TECH is the technology factor assumed to be 1.75 and $COST_{FY99}$ is the unit flyaway cost per aircraft in FY 99 dollars. All such dollar conversions were accomplished using Consumer Price Index information provided by the United States Bureau of Labor Statistics.

3.6.3.2 Flyaway Cost Estimate By Analogy with C-5 and C-17. The Team assumes the conceptual BWB aircraft design for a heavy lift aircraft is 75% more complex than the current C-5 design. This is based on the assumption of advanced technology composite materials and the advanced, non-circular fuselage structural design of a BWB aircraft. Direct estimate by analogy with the C-5 design yields:

$$F(C5)_{FY96} = 1.75*(\$184.2M) = \$322M$$
$$F(C5)_{FY99} = F(C5)_{FY96}*(1.645/1.545) = \$322M*(1.0647) = \$342.8M$$

where $F(C5)_{FY96}$ equals the unit flyaway cost per aircraft in FY 96 dollars and $F(C5)_{FY99}$ equals the unit flyaway cost per aircraft in FY 99 dollars based on the C-5 analogy. The C-5 flyaway cost analogy is based on a production of 126 aircraft. Based on the assumed BWB acquisition of 100 aircraft, this equates to a BWB Total Aircraft Acquisition Cost (TAAC)of:

$$TAAC = F(C5)*100 = \$342.8M*100 = \$34.28B$$

where TAAC is defined in FY 99 dollars.

Direct estimate by analogy with the C-17 Design yields:

$$F(C17)_{FY96} = 1.75*(\$180M) = \$315M$$
$$F(C17)_{FY99} = F(C17)_{FY96}*(1.645/1.545) = \$315M*(1.0647) = \$335.4M$$

where $F(C17)_{FY96}$ equals the unit flyaway cost per aircraft in FY 96 dollars and $F(C17)_{FY99}$ equals the unit flyaway cost per aircraft in FY 99 dollars based on the C-17

analogy. The C-17 flyaway cost analogy is based on a production of 120 aircraft. Based on the assumed BWB acquisition of 100 aircraft, this equates to a BWB Total Aircraft Acquisition Cost (TAAC) of:

$$TAAC = F(C17) * 100 = \$335.4M * 100 = \$33.54B$$

where TAAC is defined in FY 99 dollars.

This estimate by direct analogy provides a first "guess" of aircraft cost, but is very subjective in estimation of aircraft complexity compared to an existing aircraft. More detailed cost estimates are possible by using linear regression "trends" in historical aircraft costs.

3.6.3.3 RAND Cost Estimating Relationships for Aircraft Airframe Only. The following CER's based on linear regressions of historic military aircraft costs were applied by the Team to the conceptual aircraft design in this effort:

Aircraft Airframe CER for All Mission Types (Hess, 1987a):

$$\mathrm{T} = 2.57 * \mathrm{EW}^{0.798} * \mathrm{MA}^{0.736} = 2.57 * 436{,}523^{0.798} * 547.7^{0.736} = \$8.437M$$
$$T_{actual} = \$8.437B$$
$$T_{unit} = \$84.37\mathrm{M}$$
$$\mathrm{T}_{\mathrm{FY96}} = \$84.37 * (1.545 / 0.58692) = \$222M$$
$$T_{FY99} = {*T_{FY96}} * (1.645 / 1.545) = \$222M * (1.0647) = \$236.4M$$

where T equals the total program acquisition cost for a total of 100 aircraft (in thousands) of FY 77 constant dollars. EW equals the empty weight of the aircraft in pounds, MA equals the maximum airspeed of the aircraft in knots, and T_{actual} equals the total program acquisition cost for one hundred aircraft in FY 77 dollars, instead of thousands of dollars. T_{unit} equals the unit total acquisition cost per aircraft in FY 77 dollars. T_{FY96} equals the estimated unit total acquisition cost in FY 96 dollars and T_{FY99} is the estimated unit total acquisition cost in FY 99 dollars. To compose a flyaway cost estimate for the conceptual

aircraft, an estimate of the avionics and engine costs must be added to the calculated cost above for airframe only.

3.6.3.4 Raymer Aircraft Avionics Cost Estimate. Raymer states that aircraft avionics cost typically between five and twenty percent of the airframe cost (Raymer, 1989). The Team estimated avionics for the conceptual transport BWB aircraft to be 20% of the airframe cost. This is because the BWB design concept has a new set of control laws required to guide the flight of such an unconventional aircraft. This may necessitate special flight control software. The avionics cost estimate based on the RAND airframe cost estimate found in the section above is:

$$AV = 0.2 * \$236.4M = \$47.3M$$

where AV equals the avionics cost estimate in FY 99 dollars.

3.6.3.5 C-17 Analogy Aircraft Engine Cost Estimate. The cost of the BWB engines were estimated parametrically by comparison with the C-17A, the most recently produced USAF aircraft. The cost of each of its four engines is \$2.5M. The BWB transport concept was assumed to also have four engines, where N is equal to the number of engines required, but the technology and manufacturing processes required to design the engines and integrate them with the BWB itself will be very different from those used for transport aircraft today. The team assumes a technology factor, TECH, equal to one and a half, therefore, the engine cost estimate is:

$$EN = N * \$2.5M * TECH = 4 * \$2.5M * 1.5 = \$15M$$

where EN is the estimated engine cost in FY 99 dollars.

3.6.3.6 BWB Flyaway Cost Estimate. Flyaway cost for the BWB design is determined by summing the costs for airframe, avionics and engines. The flyaway cost was estimated based on the RAND airframe cost, the Raymer avionics estimate and the

C-17 analogy engine estimate from sections 3.6.3.3 through 3.6.3.5 above. Therefore, the flyaway unit and total program cost estimates are:

$$FL_{unit} = T_{FY99} + AV + EN = \$236.4M + \$47.3M + \$15M = \$298.7M$$
$$FL_{TotalFY99} = FL_{unit} * 100 = \$29.87B$$

where the unit flyaway cost estimate FL_{unit} and $FL_{TotalFY99}$ are expressed in FY 99 dollars. The RAND estimated unit flyaway cost is higher than the analogous cost estimates based on the C-5 and C-17 aircraft, however, the BWB design represents a major departure from the standard aircraft in terms of manufacturing and structural loading and this estimate may be more realistic than analogy with currently existing aircraft.

3.6.3.7 Large Subsonic Transport Aircraft Analogy O&S Cost Estimate. In order to estimate the O&S cost for a new BWB design aircraft it is first necessary to obtain an estimate of the average number of flying hours per year for the aircraft. An average of the actual flying hours per year for the C-141, C-5 and C-17 aircraft in the United States Air Force was obtained from the Air Force Total Ownership Cost web page (AFTOC, 2000). The average number of flying hours over five USAF transport aircraft (C-5A, C-5B, C-141B, C-141C and C-17A) was 67,000 hours per year. Therefore, the flying hours per year, or FH, was assumed for the new fleet of BWB aircraft to be 67,000. The O&S cost for a new plane was based on the actual costs incurred for these previous Air Force large, subsonic, transport aircraft. The actual costs were also gleaned from the AFTOC web page and averaged over various Air Force Major Commands. The average annual O&S cost per year was estimated from actual FY 1999 costs for the above mentioned USAF aircraft to be \$11,700 per flying hour. The current annual and total O&S cost to operate Air Force heavy lift aircraft at the assumed average rate of \$11,700 per flying hour over the assumed average number of annual flying hours for a thirty year

expected aircraft is:

$$O\&S_{CurrentAnnual} = \$11{,}700 * 67{,}000 = \$783.9M$$
$$O\&S_{CurrentTotal} = O\&S_{CurrentAnnual} * 30 = \$2.352B$$

where the above cost estimates are made in FY 99 dollars.

Based on initial estimates of nineteen percent fuel efficiency gains possible with the BWB design, the average O&S cost for the BWB aircraft is assumed to be eighty-one percent of that incurred today. The calculated BWB operating cost (OC) per flying hour is thus:

$$OC = \$11{,}700 * 0.81 = \$9{,}477$$

where OC is expressed in FY 99 dollars. Therefore, the total estimated annual and total O&S cost for the Team's conceptual heavy transport aircraft is:

$$O\&S_{BWBannual} = OC * FH = \$9477 * 67{,}000 = \$635M$$
$$O\&S_{BWBtotal} = O\&S_{BWBannual} * 30 = \$635M * 30 = \$19.05B$$

where $O\&S_{BWBannual}$ is the annual operating cost for the 100 unit BWB aircraft fleet in FY 99 dollars and $O\&S_{BWBtotal}$ is the total O&S costs over the thirty year expected BWB aircraft service life in FY 99 dollars.

Using the efficiencies of the BWB design could realize a projected savings in total O&S costs of:

$$O\&S_{Savings} = O\&S_{CurrentTotal} - O\&S_{BWBtotal} = \$23.5B - \$19.05B = \$4.45B$$

in FY 99 dollars over the conventional transport design.

3.6.3.8 Cranfield University Estimate of BWB Costs. The cost estimates contained within the Cranfield University study are included in this section to provide a comparison with the RAND parametric cost estimate developed by the Team. The

Cranfield study was performed under different assumptions than this effort. The Direct Operating Costs (DOCs) of an aircraft are highly dependent on the initial acquisition cost since the airline has to deal with aircraft depreciation and finance payments. Thus, a reasonable estimate of aircraft acquisition cost was first obtained. Overall, the different cost-estimation methods used by Cranfield University for unit acquisition cost showed that their conceptual design, the BW-99 aircraft, is estimated to cost $164M, for a production run of 100 aircraft.

This estimate is almost 50% less than RAND estimated cost estimate for a BWB transport aircraft. However, this is due to the fact that the Cranfield University BWB study was performed under different aircraft sizing assumptions. The cost calculation showed that the unit acquisition price was highly dependent on the price of the engines and of avionics. As mentioned in section 2.4.2, for a reasonable range of aircraft acquisition costs, the BWB design represents a savings of 10-19% in DOC's when compared to the Boeing 747-400. These encouraging figures for unit aircraft acquisition cost and cost of operation reflect the potential gains to be won by pursuing non-circular fuselage aircraft designs such as the BWB.

3.6.3.9 Large Subsonic Transport Aircraft LCC Estimate. The LCC is composed of the cost of the total aircraft program acquisition cost including flyaway, O&S and disposal costs. Flyaway cost includes aircraft airframe, avionics, engines, and everything else that composes the aircraft itself. Table 3-7 summarizes the cost estimates performed and compares the BWB flyaway cost predicted for the Team's conceptual aircraft design with three other independent flyaway cost estimates. Table 3-7 summarizes the calculations required to produce the LCC of the BWB aircraft design.

Table 3-7 BWB Annual Cost Estimates in FY 99 Dollars

Airframe Cost – RAND Study	$236.4 M
Avionics Cost - Raymer	$47.3 M
Engine Cost – C-17 Analogy	$15 M
BWB Flyaway – Sum of Above Three Estimates	$298.7 M

Table 3-8 Annual Flyaway Cost Estimates For Comparison With BWB

Type of Cost Estimate	Cost Estimate (in FY 99 Dollars)
Flyaway Cost - Raymer	$226.1 M
Flyaway Cost – C-5 and C-17 Analogy	$338.0 M
Flyaway Cost – Cranfield University	$164 M

Table 3-9 Lifetime Cost Estimate for BWB Design Aircraft in FY 99 Dollars

BWB Flyaway Estimate Per Aircraft	$298.7 M
BWB Fleet O&S Estimate Per Year	$635 M
Total BWB Flyaway For 100 Aircraft	$29.87 B
BWB Fleet Total O&S Cost Over 30 Years	$19.05 B
BWB Total LCC Over 30 Years – Sum of Above Two Items	$48.92 B

Therefore, the Team's estimate of LCC is composed of:

$$BWB_{LCC} = FL_{TotalFY99} + O\,\&\,S_{Total} = \$29.87B + \$19.05B = \$48.92B$$

where the LCC cost is given in FY 99 dollars.

3.7 Decision Making

Once the first iteration of analysis and optimization has been performed, the results need to be interpreted. Decision Making is the process by which the alternative is scored according to the system devised in the Value System Design. The scoring of the alternative should give ideas for improving the performance of the alternative generated

in the next iteration. For the team's process, a scoring function was not established, however, meeting the sponsor's satisfaction was equated with success

The design of the optimized wing was presented to the sponsor for evaluation. The sponsor determined that the loading for FEA was oversimplified by applying one point loading at the wing tip to model the force of the fuselage on the wing. This unrealistic loading caused the simple model (with no rod reinforcement members) to twist and deform in an undesirable and unrealistic way. Therefore, in order to obtain a more realistic model, the Team decided to follow the sponsor's advice and change the FEA loading appropriately. and reinforcement rod elements were added.

3.8 Implementation

Ideally, the design team should use the information gathered from the previous value system design iteration to determine whether to stop the design process, or reiterate. The stated desire of the sponsor was for the second iteration of the process to generate an aircraft model which better represented a BWB design. The steps taken to develop the new model are described in previous sections.

As stated above in section 3.7, a change in the FEA aircraft loading model was urged by the sponsor. Therefore, the following FEA loading changes were implemented and analysis and optimization of the aircraft model were repeated. The point loading representing the force of the fuselage on the wing that was previously applied only at the wing tip was now divided evenly into sixteen parts and applied at 8 nodes along the center of the top wing skin and the corresponding 8 nodes along the bottom skin. The point loading representing wind drag force that previously was applied at the wing tip

was now divided into twenty-eight parts and applied at 14 nodes along the top leading edge of the structural wing box and 14 nodes along the bottom leading edge of the wing box. Figure 3-25 below illustrates the top wing skin loading changes that the Team implemented (similar loadings were applied to the bottom wing skin). Following this implementation, the sponsor determined that both models yielded feasible optimal solutions based on the given constraints and was satisfied with the results of the Team's process based on the two simple models created.

Possible avenues for continuing efforts beyond the thesis are discussed in Chapter 5.

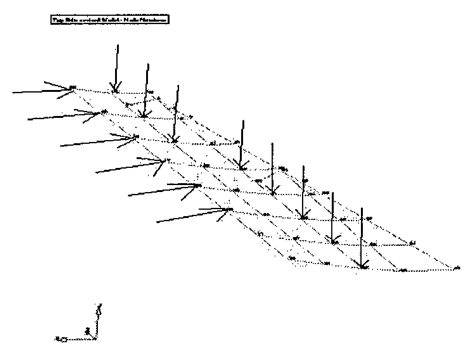


Figure 3-25 Re-distributed Forces Applied to FEA Model

4. RESULTS

4.1 Thesis Requirements

The results of the thesis must be defined relative to the initial requirements levied by the thesis sponsor. The sponsor defined the initial thesis requirements in terms of expected results. The following results were expected for the conceptual aircraft model and associated process developed within this effort:

1. Create a conceptual aircraft model within AML that is simple, flexible and easy to change (capable of rapid change from conventional Boeing 777 type design to a BWB design as well as intermediate designs).

2. The AML model must be capable of being geometrically meshed with quadrilateral elements (for finite element analysis) using the AML to MSC.PATRAN interface within AML.

3. Connectivity files containing node locations and element definition must be generated during meshing within AML.

4. Demonstrate conversion from the mesh connectivity files generated by AML to a form acceptable for finite element and other analysis using the ASTROS software.

5. Analyze the entire aircraft structure using ASTROS, including static loading deformation, element stress under loading, optimization of structural weight and member thickness, modal, flutter and aerodynamic analyses.

6. Change the aircraft design model from a conventional type aircraft to a BWB design (demonstrating model flexibility).

7. Regenerate connectivity files for subsequent ASTROS analysis run on the BWB design.

8. Analyze the entire new BWB aircraft structure using ASTROS, including loading deformation, element stress under loading, optimization of structural weight and member thickness, modal, flutter and aerodynamic analyses.

9. Demonstrate that the AML model reduces iteration time (Time required for model creation on second iteration must be less than time required for first iteration).

10. Establish, demonstrate and document the over-arching process to accomplish all the above items.

4.2 Thesis Requirements Fulfillment

The team sought to fulfill the sponsor defined, thesis requirements stated above. As illustrated in Table 4-1, the Team completed seven requirements and partially completed the three remaining requirements. The thesis sponsor was satisfied with the results of the thesis effort.

Table 4-1 Thesis Requirements Fulfilled By the Team

Number	Requirement	Completed?
1	AML Model	Partially
2	PATRAN Meshing	Yes
3	AML Meshing Files	Yes
4	ASTROS Input Files	Yes
5	Full ASTROS Analysis	Partially
6	Change to BWB Design	Yes
7	Recreate ASTROS Input Files	Yes
8	Full ASTROS Analysis Again	Partially
9	AML Model reduces Iteration Time	Yes
10	Establish Overall Process For 1-9	Yes

The main requirement for the thesis was to develop an aircraft conceptual design process which could rapidly model a new concept design and produce analysis results for use in design iteration. The tools specified for use in the design process had never before been integrated to produce the results achieved by the Team. However, many software

delays affected the initial process formation. This was due to the fact that many of the AML capabilities required by the thesis effort did not exist or were unproven when the Team began work. The thesis effort required the use and integration of software objects and methods which were not yet fully operational. Below are each of the thesis requirements and the degree to which they were fulfilled by the Team.

4.2.1 Construct a Simple, Flexible Aircraft Model Using AML. The requirement to develop a flexible AML aircraft model was partially completed and the software code is included in Appendix C. The simple AML aircraft model constructed by the Team encompasses many of the original goals, but time did not allow inclusion of aircraft engines, landing gear and other more complex aircraft modeling details. The AML model is compatible with PC or UNIX computers. The underlying AML aircraft model based on an eight point, two-dimensional planform, or outline, is simple and easy to change yet powerful enough to support a variety of unconventional, non-circular fuselage aircraft designs. Examples of two radically different aircraft designs possible using the same initial AML model are shown below in Figures 4-1 and 4-2.

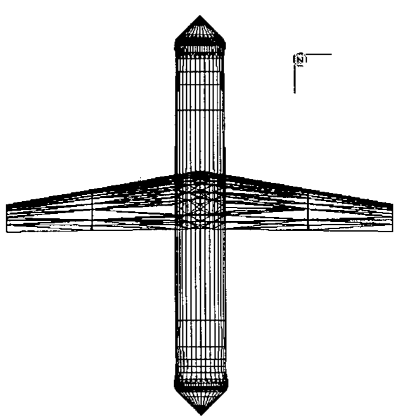


Figure 4-1 Conventional AML Model

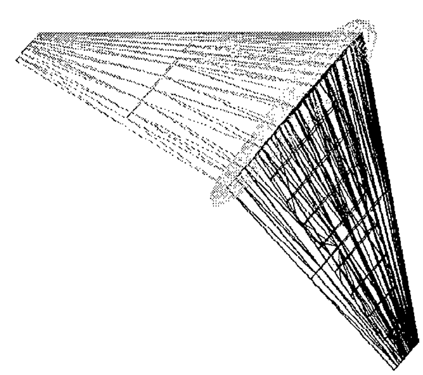


Figure 4-2 BWB AML Model

The AML model relies on only 24 variables to generate the aircraft design. The change from Figure 4-1 to Figure 4-2 was performed in less than one hour by adjusting only 10 of the 24 possible variables. Volumetric analysis of the conceptual aircraft design was made possible through the use of the cargo objects to test payload requirements against the projected payload area within the concept aircraft design.

4.2.2 Geometric Model Meshing Using Quadrilateral Elements. The requirement to develop a geometric finite element mesh for the model composed of quadrilateral elements was completed using the AML to MSC.PATRAN interface. This interface is only available using UNIX AML version 3.1.3 or later. The meshing capability using quadrilateral elements via the AML to MSC.PATRAN interface has never before been accomplished and was a major goal for the sponsor. The thesis provided a very tangible result by implementing this process. Working extensively with the software manufacturer to implement this new technology process aided greatly in smoothing out many of the software wrinkles encountered.

The AML to MSC.PATRAN interface allows the AML model to be sent to MSC.PATRAN for geometric meshing using quadrilateral elements. The interface is transparent to the AML user and the MSC.PATRAN results are returned to the user in AML in the form of a series of grid location and connectivity files. These files are now suitable for incorporation into an FEA software input deck, or file. The files contain information about the grid locations of each node as well as node connectivities defining each element.

4.2.3 Convert AML Meshed Connectivity Files to a Form Acceptable for Analysis Using ASTROS. This requirement was completed by the thesis team once in each design

iteration for the conventional wing-box structure and again for the blended wing body structure. This process was very tedious and involved taking raw data and converting it into an ASTROS compatible format. The work of debugging the ASTROS input deck was accomplished by working closely with expert personnel in AFRL/VA.

4.2.4 Use ASTROS to Perform Design Analysis On the Entire Conventional Wing Aircraft Structure Including Structural, Aeroelastic, and Weight Optimization Analyses. This requirement was partially completed by the Team. Time did not allow the entire conventional wing aircraft structure to be completed in the AML model. The fuselage structure was not added to the conventional aircraft structural model. A simplifying assumption was made and the conventional wing box alone, not including any of the conventional fuselage structure, was evaluated using ASTROS. The structural analysis and weight optimization analysis of the conventional wing box model was completed, but time did not allow the other analyses possible within ASTROS. The ASTROS output results obtained included deformation under loading, member stresses under loading and optimization of structural member sizing (weight). The initial, user-supplied, "guessed" weight of the conventional wing structure began at 5,800 lbs. This initial weight was in the infeasible solution region because it did not meet the maximum allowable stress constraints imposed on the ASTROS optimization given the wing loading. ASTROS performed eleven optimization iterations and converged on an optimal minimum wing structural weight of 10,700 lbs. The optimal solution was in the feasible solution region and met the maximum stress constraints for the structure. The results were given to the sponsor for quality check and deemed acceptable as a feasible wing structure design based on the given loading. Time did not allow further ASTROS analysis such as aero-

elastic, flutter and modal analyses. They are recommended for further study in Chapter 5.

4.2.5 Rapidly Change the Aircraft Model from a Conventional Aircraft Design to a Blended Wing Body Design. This requirement was fulfilled by the thesis team. The thesis sponsor recommended a second aircraft design modeling iteration using an elongated BWB design. The original AML model of a conventional aircraft and wing-box structure was changed to a non-circular fuselage BWB design. The change from conventional wing design to BWB design was performed in less than one hour by adjusting only 10 of the 24 possible variables. Figure 4-2 above illustrates the changes apparent in the new BWB design. Again, using the AML to MSC.PATRAN interface discussed previously, the BWB design was assigned a geometric mesh composed of quadrilateral elements. The same previously mentioned node grid location and element connectivity files were again generated within AML for the BWB design and converted into a form acceptable for ASTROS analysis.

4.2.6 Use ASTROS to Perform Design Analysis of the entire BWB Aircraft Structure Including Structural, Aerodynamic and Weight Optimization Analysis. As was the case with the conventional wing-box structure described in section 4.2.4 this requirement was only partially completed. The entire BWB aircraft structure was assumed to be simply two joined wing box structure halves joined in the middle (see Figure 4-2 above). The structural analysis of the model was completed, but time did not allow the other analyses possible within ASTROS such as modal, flutter and aerodynamic analysis. The ASTROS output results obtained included deformation under loading and optimization of structural member sizing (weight) for the BWB design.

The user-supplied, first "guessed" weight of the BWB wing structure began at 18,500 lbs. This initial weight was in the infeasible solution region because it did not meet the maximum allowable stress constraints imposed on the ASTROS optimization given the wing loading. ASTROS performed 11 optimization iterations and converged on an optimal minimum wing structural weight of 28,400 lbs. The optimal solution was in the feasible solution region and met the maximum stress constraints for the structure. The results were given to the sponsor for quality check and again deemed an acceptable and viable aircraft design optimized for structural weight. Time did not allow Further ASTROS analysis for the BWB design such as aero-elastic, flutter and modal analyses and they are recommended for further study in Chapter 5.

4.2.7 Demonstrate That the AML Model Significantly Reduces Aircraft Conceptual Design Iteration Time. Table 4-2 illustrates the time reduction in certain areas of the aircraft conceptual design process which occurred as a result of using the AML parametrically defined model. The development time required to produce a model decreased from three months for the first iteration to less than one day for the second iteration. Future changes to the AML model should be similarly accelerated. This final requirement is very significant to the thesis sponsors. Reducing iteration time allows more possible aircraft designs to be considered, expanding the design space and enhancing the aircraft design process.

Table 4-2: Time to Complete Actions In Iterations 1 and 2

Action	Iteration 1	Iteration 2
AML Model Development	3 months	2 hour
Mesh Creation	1 day	2 hours
Formation of ASTROS Input File	2 days	1 day
ASTROS analysis	2 hours	2 hours
ASTROS optimization	2 hours	2 hours

4.2.8 Establish, Demonstrate and Document the Process That Accomplishes All the Above Items. The process to accomplish all the above items was established by the thesis team in the time and effort spent to complete each of the thesis requirements. The process has been captured for the benefit of the thesis sponsor as well as for future efforts that may continue this work. Figure 4-3 below illustrates the data flow for the conceptual aircraft design process developed by this effort.

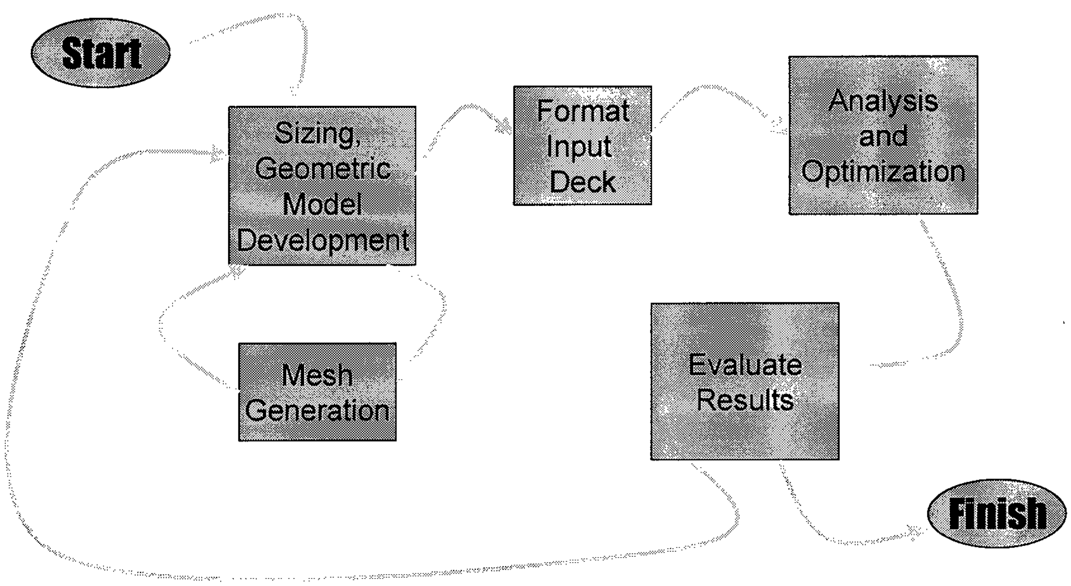


Figure 4-3 Software Process Diagram For Systems 2000 Aircraft Design Process

5. CONCLUSIONS & RECOMMENDATIONS

5.1 Conclusions

The Systems Engineering Team designed a comprehensive process in which a flexible, parametrically driven aircraft model was developed with a minimum of variables. The process was completed with the model successfully meshed and optimized by a finite element analysis program.

The ability for the subobjects of the aircraft model to inherit characteristics from one another and their ancestors is particularly powerful. Relatively detailed models can be created from the Team's code, and these models can be rapidly changed. The demand-driven environment of AML allows for changes to be performed interactively, if desired, with the effects of parameter changes being visible to the user. The nature of the objects is such that all or part of them can be reused or adapted by future users.

The process's rapid turnaround of information on new designs was demonstrated by the second iteration of the software process. A radically different aircraft design was created and analyzed in little more than one day, by using the same tools and objects that were used by the first aircraft design.

The generation of a mesh acceptable to a finite element analysis program from a geometry-centered modeling language like AML is a notable achievement. While the process to get to this point has been arduous, FEM mesh generation from a parametrically driven object-oriented model is a first step for integrating optimization routines into conceptual design.

As noted in section 2.6, new software package such as UNIX AML are primarily tested for the major UNIX architectures and are often implemented across the board without regard for less popular UNIX configurations. Future problems reported from the other untested architectures are solved as they appear and, therefore, every architecture in today's UNIX environment cannot be supported by every software company. Some UNIX architectures inevitably fall through the cracks and their software problems must be dealt with after the fact. This appears to be true in the Team's experience implementing cutting edge AML software capabilities using the SGI version of UNIX AML. The Team believes there must exist a superior method to quality check software and provide software customer support. Quality of customer support cannot be ignored in the competitive world marketplace of computer software.

However, the customer support at TSI was extremely helpful and responsive in fixing the software implementation problems as they arose. The Team was able to accomplish many of the aircraft conceptual design and evaluation process goals for the thesis effort. This fact alone is a tribute to the patience and dedication of the TSI customer support team, without which this effort would not have been possible.

5.2 Recommendations for Future Research

This thesis represents the first effort at integrating a fully flexible, parametrically driven conceptual aircraft model with a geometric mesher and finite element analysis software. Considerable opportunities exist to continue the work of the Team, both in the depth of the application and the breadth of it.

The parametric model developed by the team is an extremely simple yet seemingly powerful one. Refinements in the structure of the code and the function of the code could be made. AML model details including engines and external stabilizers could be added. The AML code could be expanded to contain material and cost information; mission profile objects could be integrated; additional structures and substructures could be added, all without unduly increasing the number of defining parameters.

Outside of the AML model, the interaction of the PATRAN meshing program with AML is worthy of considerable study in and of itself. Using what is essentially a geometric mesher to generate meshes appropriate for FEA brings up the question of how to use a dumb tool smartly. Ideally, the mesh returned by PATRAN would be in a form that could immediately be turned into input decks for ASTROS, with no need for manual sorting out of improper connectivities. The Team also recommends investigating automation of the process by which the connectivity files of each structure are transformed into input decks. In order for this to happen, methods, either within PATRAN or AML, must be developed to "train" PATRAN to provide an acceptable mesh.

Further ASTROS analysis of both conventional and BWB designs should be performed in order to document improvements which may be achieved in BWB designs. Other further capabilities of ASTROS which were not used in this effort include aeroelastic, flutter and modal analysis. Additional aircraft substructure may be required before these analyses are performed. In addition, the design space over which ASTROS optimizes is fairly small. If the generation of designs to be presented to ASTROS were automated, a fuller appreciation of the design space could be realized.

Work can also be performed on the systems engineering process itself. The range of possibilities created by the ability to rapidly build FE models is large, and can support a systems engineering or decision analysis study of what path future research should take. Polling about what characteristics make conceptual designs "good" can drive the development of fitness functions which can in turn be used as part of an automated optimization across a wider design space than is currently allowed in ASTROS.

As the MSC.PATRAN mesh information for the model is coded into an ASTROS input file, the ability to view the FEA model in a CAD type program such as HyperMesh or MSC.PATRAN is essential. This facilitates application of FEM forces and constraints in the proper locations with a minimum of effort. The Team recommends that further efforts install MSC.PATRAN (version 8+ is currently required for the AML-to-MSC.PATRAN meshing capability) on the same cluster of machines where UNIX AML is installed. This will allow for model creation and meshing on the same machine.

The Team created the AML model at AFIT, but were forced to mesh the model on two separate occasions in Building 146 using AFRL/VASD personnel and resources because MSC.PATRAN and SGI UNIX AML were installed there on the same machine cluster. The Aero laboratory at AFIT currently has a licensed version of MSC.PATRAN 7.5 on the Digital UNIX machines with upgrades to MSC.PATRAN 9.0 possible in the future (Digital UNIX is not currently supported by TSI for AML). It may be possible to use this installation version with additional licensing costs for the SGI UNIX machines in Room 2013 where SGI UNIX AML is currently installed.

APPENDIX A: SOFTWARE TOOLS USED

A.1 The Adaptive Modeling Language (AML)

AML is an adaptive, object oriented, modeling language useful for knowledge-based concurrent engineering. It is a comprehensive modeling paradigm to integrate design specifications, part geometry/features, manufacturing, inspection and analysis processes in a unified part model. AML provides a Knowledge Based Engineering (KBE) framework that captures knowledge from the modeled domain and creates parametric models with that knowledge. Classes inheriting from AML primitives may be defined and methods may be written against these classes providing user-defined behavior.

After defining the classes, a hierarchical part model is instantiated wherein the attributes of objects can be related using unidirectional non-cyclic constraints. This part model may be utilized as a parametric design in a "what-if" scenario by changing design parameters and re-computing the model as the constraints are propagated through the model on demand. AML is "adaptive" in that it can be used to model a wide range of domains that have inter-acting components and the constrained behavior between them. Hence, it can be adapted to diverse engineering applications. In addition, AML is dimension-less and may be adapted to utilize any dimension units.

AML can be used to detail various aspects of a problem through a single unified model. Structural aircraft design is an example of such an application. A geometric design is created, followed by the association of various physical attributes with the geometry. Then the attributes for a finite element mesh and the knowledge required for

generating input analysis files is maintained. AML allows all this information to be stored in a structured fashion within a single model. Furthermore, knowledge for manufacturing, inspection and tooling can be incorporated in the same model for the automation of manufacturing and inspection process plans. Feedback may be provided at various stages to different entities in the model. A complete user interface for the problem including input forms, output forms and menus can also be associated with the same part model encompassing the various aspects of the application.

The first step in building a rapid aircraft modeling process involved constructing a parametrically defined aircraft model within AML. TechnoSoft, Incorporated (TSI) supplied the Air Force Research Laboratory's Air Vehicles Directorate (AFRL/VA) with a CDROM containing the Adaptive Modeling Language (AML) software while under contract. TSI also later provided the download Internet site in order to obtain a UNIX version of the AML software. AFRL/VA sponsored the AFIT GSE design study and possessed a transferable AML license through TSI. Through the existing license, it became possible for AML to be installed on the three PC computers in the AFIT Systems Engineering Room, Room 149C, Building 640, Area B, Wright Patterson Air Force Base, OH. The PC version of AML was also installed on each team member's home PC computer to allow effort to proceed from the home offices. The UNIX version of AML was later installed on an SGI UNIX machine at AFIT in room 2011 of building 640.

A.2 MSC.PATRAN

MSC.PATRAN is an open-architecture, general purpose, three-dimensional Mechanical Computer Aided Engineering (MCAE) software package with interactive graphics providing a complete CAE environment for linking engineering design, analysis and results evaluation functions. It provides a complete software environment for companies performing simulation of mechanical products. MSC.PATRAN is produced by the MSC.Software Corporation. MSC.PATRAN is a leading finite element modeler that enables the user to conceptualize, develop and test a product using computer-based simulation prior to making manufacturing and material commitments. Major manufacturers around the world use MSC.PATRAN as the basis for their product improvement process, reducing or eliminating costly physical prototyping and testing.

By using MSC.PATRAN, engineers can create finite element models from their computer-aided design (CAD) parts, submit these models for simulation, and visualize the simulated model behavior. The results are then used to improve their product designs to better resist operating loads, reduce weight or material, or have higher performance. MSC.PATRAN has great breadth and depth of CAE functionality. It supports all leading CAD software and analysis software programs. The software is fast, easy-to-use and highly customizable, enabling engineers to create their models quickly and directly incorporate MSC products into their specific engineering processes (PATRAN, 2000)

MSC.PATRAN was used within the thesis effort only via the AML to MSC.PATRAN interface to generate a geometric model mesh. The interface is transparent to the AML user and the MSC.PATRAN software graphical user interface is never actually engaged. The AML to MSC.PATRAN interface must occur on a UNIX

machine (or an X-Windows type environment established with such a UNIX machine) which is itself on the same cluster of machines where MSC.PATRAN is resident. TechnoSoft, Incorporated is working with the AML to MSC.PATRAN interface to expand the remote access to PATRAN capability. Many more details regarding the powerful capabilities of MSC.PATRAN are available at the MSC web page address (MSC.Patran, 2000).

A.3 Automated Structural Optimization System (ASTROS)

ASTROS is a comprehensive FEA software package designed for the United States Air Force Research Laboratory's Flight Dynamics Directorate by Silicon Graphics, Incorporated. It is considered the lead aeronautical structural analysis program in the world. It has been used successfully to analyze aircraft systems such as the F-16 fighter in the past. ASTROS was recommended by Dr. Vipperla Venkayya, the project sponsor in AFRL/VA, as the state-of-the-art aeronautical design analysis package which is sufficient for use in this effort. The rights to the ASTROS software were recently purchased by the MSC.Software Corporation. For comparison, MSC.NASTRAN, a commercially available FEA software currently marketed by the MSC.Software Corporation performs all the same functions as ASTROS with certain additional functions. ASTROS is still available for commercial use, but its marketing future is unclear.

ASTROS can operate under two different boundary conditions known as Analyze or Optimize. Many different types of analyses, or disciplines can be performed during a

given boundary condition. Structural analysis, element displacements and element loading stresses may be analyzed under the Analyze boundary condition. Structural member thickness and weight may be minimized for a given loading under the Optimize boundary condition. It is this multidisciplinary capability that makes the ASTROS code viable in a preliminary design context. Neill and Herendeen state that ASTROS is capable of performing analyses in the following areas:

- Static Structural Analysis
- Normal Modes of Vibration
- Steady-State Aeroelastic Analysis
- Aeroelastic Stability Analysis of Flutter
- Transient Response Analysis
- Frequency Response Analysis
- Transient response to a Nuclear Blast
- Non-planar Rigid Static Aerodynamic Analysis

Additional detail on the many capabilities of ASTROS may be found in the full text of the ASTROS User's Manual, the ASTROS Application Manual, the ASTROS Programmer's Manual or the ASTROS Theoretical Manual. (Neill, 1995)

APPENDIX B: SOFTWARE DESIGN PROCESS DETAILS

AML 3.1.2 (SGI UNIX version) Installation And Setup Process

This section describes the process followed to acquire, install and utilize the SGI UNIX version of AML for aircraft modeling.

Background: The parametric aircraft model was constructed using the PC computer version of AML 3.1.1. The project sponsors specified that structural FEA of the aircraft model must be accomplished using quadrilateral finite elements. AML version 3.1.1 for PC computers only supports a triangular element model meshing capability. It was then discovered that the UNIX version of AML 3.1.2 allows meshing of a model using quadrilateral elements for FEA analysis via the AML to MSC.PATRAN interface. UNIX AML 3.1.2 was the first available version of AML that provides a direct interface to the UNIX meshing software known as MSC.PATRAN for the purpose of model meshing and assignment of quadrilateral element connectivities. The SGI UNIX version of AML was subsequently downloaded from the TechnoSoft, Incorporated. (TSI) customer support Internet web page (www.technosoft.com) using log in information provided by the company.

A great deal of time was expended learning about the UNIX version of AML, what platform and software versions were essential to running the SGI AML 3.1.2 version as well as how to acquire an account on a UNIX machine where a licensed version of MSC.PATRAN resides or may be accessed. The Team recommends working directly with TSI to ensure the target UNIX machine meets the graphics and other requirements to be a "good" candidate on which to install the UNIX version of AML.

After some research, it was discovered that MSC.PATRAN resided on a Digital cluster of UNIX machines in the AFIT Aeronautical Department laboratory, room 129. A UNIX account was obtained on the Digital cluster of Unices in room 129 because this would make it very simple to use AML and us the AML to MSC.PATRAN interface to call to the machine where MSC.PATRAN resided. However, it was soon discovered that TSI does not support Digital UNIX machines for any version of AML. Therefore, an account was established on an adjacent SUN cluster of machines in the same room. After downloading and installing the SUN UNIX version of AML, an appropriate key file from TSI was obtained after checking to ensure the proper graphics suite was resident on the machine. After a week long struggle with the license server, it was concluded that a faster SGI UNIX machine would be required to more conveniently run AML.

After a brief AML license server problem on the new SGI UNIX machine was resolved with the aid of the local UNIX system administrator, guidance was required on how to actually operate using the UNIX version of AML 3.1.2, which is slightly different from the PC Windows based version of AML 3.1.1. TSI promptly provided this guidance. The following steps outline the process used to access and use UNIX AML.

1) The team logged in on the SGI UNIX machine in room 2011, which uses the IRIX 6.5 operating system, and opened a UNIX command terminal. Navigation was accomplished from the default home directory to the "TechnoSoft" directory in which AML resides.

2) The command “AML” was required at the UNIX command line to start the AML software. At this point, a “Xemacs” command and editing window is opened and a loading process occurs wherein the AML license file is accessed. After some time the “AML >” command prompt is displayed. This is one of the two UNIX AML modes, the AML command prompt mode. To reach the second AML mode, the graphics mode, one must use the “F6” command button along the top of the computer keyboard. The “F6” button switches UNIX AML from the AML command line mode to the graphics, or Graphical User Interface, mode. The graphics mode in UNIX AML is very similar to the graphics layout toolbars in the PC version of AML.

3) To edit an AML file, open the file in the Xemacs window (From the "File" menu, "Open" is an option. Alternately, hit the “Alt” and “x” keys simultaneously (called <Meta-x>) to get to the Xemacs command line, and type "open" and hit return to follow the process.

4) The file will then be loaded into a new Xemacs buffer and should appear in one of the two buffer screens. Changes can be performed on the file and files can be saved in a process similar to how they were opened using the “File”, then “Save” pull down menu in the Xemacs window.

5) To load an AML file into the graphics layout editor, hit <Meta-x> to get to the Xemacs command line, type "load-", and hit <Enter>. One of the Xemacs windows will display the list of possible completions (commands that begin with 'load-'). Select the one named 'load-file'. The system will then display the current directory-path and the files in the directory. Select the file to be loaded;

the AML buffer should respond with a message like "*filename* loaded into AML". To switch to the graphics layout editor, hit <F6>. In the graphics windows that open, locate the window to the right of the black graphics display window. From this window, select "New Model", then "Create Model". Type "afit-airplane-object". From the same window previously mentioned, select "Tree" to redraw the model tree. From the Model Tree window, the object and subobjects can be inspected, modified, or drawn.

6) Loading the AML to MSC.PATRANAML-to-PATRAN interface: e of AML to MSC.PATRAN interface object to mesh an AML model using MSC.PATRAN resident on another UNIX machine. First, obtain the aml-patran-interface software from TSI and copy the required files to the proper directory within the "TechnoSoft" directory. One must then change the settings in the UNIX "logical.paths" file to point to the aml-patran-interface. One must also add a line in the "logical.paths" file which points to the directory where MSC.PATRAN resides. See the example "logical.paths" file in Appendix C: Software Deliverables. From the "AML>" prompt, type "(load-system aml-patran-interface)". An error may appear and AML will again give the "AML>" prompt, but this is to be expected. Type in ": continue" and AML should finish loading the AML to MSC.PATRANAML-to-PATRAN interface.

7) Using the AML to MSC.PATRAN interface to mesh objects in AML: Objects to be meshed must first be tagged. When in the graphics portion of AML with the object to mesh open, instantiate an object of class "tagged-*modeltype*-object" where *modeltype* represents the class of object to mesh. Then instantiate an

object of class “patran-mesh-object” and then use the “Meshing” pull down window on the widest of the AML windows which may be behind the main graphics windows. The “Meshing” pull down menu contains steps for selecting the object to mesh (tagged-*modeltype*-object in this case). Then continue to click the meshing buttons in order, generate mesh, load-mesh, draw mesh. When the interface to MSC.PATRAN proceeds as planned, the object mesh may be drawn and the meshing refined by adjusting the tagged element properties.

Steps to generating connectivity files out from the afit-airplane-object

The current version of afit-airplane-object contains code which automatically generates a meshed version of the wing-box and can generate the connectivity files for each surface.

1. Ensure that the morphing system is loaded; ensure that the afit-airplane system is loaded & compiled, if needed

2. Create an instance of the afit-airplane-object

3. From the Editing Layout in AML, expand the afit-airplane-object to show the top level of subobjects

4. Draw any physical objects which help the user visualize the current plane configuration (planform, wings, fuselage, wing-boxes)

5. Draw the wing-box. This is the object which will be meshed and have its connectivity files generated. Clear the drawing screen.

6. Expand the following objects, to show their component subobjects: box-sides-series-mesh-querys, rib-series-mesh-querys, spar-series-mesh-querys.

AML will write files associated with the mesh to a folder named the same as the meshed object (in this case, WING-BOX-MESH.) Demanding a drawing of the individual subobjects will write the connectivity files to this folder. Each subobject (in this case, each rib, spar, and each side of the outside skin) will have its own set of files, detailing the nodes (0-D) and quadrilateral connectivity (2-D).

The pathway used by AML to place this directory is specified in logical.path by the :meshes line and can be determined from the AML command prompt with the command (logical-path :meshes). If no :meshes line exists, the mesh file defaults to the users' AML default directory.

The first object that is drawn will spawn MSC.PATRAN and will develop the meshed object. This can take a considerable amount of time. Successive mesh-query objects will draw almost immediately.

DRAW all the children of the box-sides-series-mesh-querys, rib-series-mesh-querys, and spar-series-mesh-querys objects.

As the subobjects are drawn, INSPECT each subobject, noting the value of the parameter, "File connectivity." This will be the leading characters of the filenames containing the node and connectivity information. The value should be similar to "001_2_2", "002_2_2", etc. (Numberings are unique to each AML session; an object which is remeshed several times may have files start off with "097...". Each time a new mesh for a particular object is created, AML writes over the previous meshes; only the most current mesh for each meshed object exists at the file location noted in logical.paths :meshes line.)

Since the filenames are numbers, it is imperative that the user keep track of which numbers reference which subobject.

There are additional files in the folder other than those needed. In order to generate a FEA input deck for a program such as ASTROS, the user needs the 2-D connectivity files (with extensions .tri3 and .quad4) and the file named wing-box-mesh.crd. The wing-box-mesh.crd file contains a list of the node numbers, followed by their x, y, and z locations in the AML model. The individual .tri3 and .quad4 files contain a column of unique (across the meshed object) element numbers, and 3 or 4 columns listing the node numbers associated with the element. Extraneous columns exist in the .quad4 and .tri3 files. Information from the files can be extracted via using Excel or any text editor.

Excel was used by the Team due to its good column-handling ability, and its ability to save spreadsheets in a column separated value format. The process of eliminating the extraneous data from a file or series of files is an obvious candidate for computerization.

Comments on the Process:

It took a great deal of effort to finally get UNIX AML to work in the circumstances it was required. The team first obtained a UNIX account on an older SUN cluster of machines. The SUN UNIX machines had the required graphics hardware capabilities specified by TSI for AML use, but were very slow (75 MHz) and did not have the latest version of OPENGL, a graphics software package required for AML use. A great deal of time was expended attempting to set up a license server for AML on the SUN server machine and run AML for the first time.

After experiencing difficulties, another UNIX system administrators at AFIT advised the team to move the AML software operation to an SGI suite of UNIX machines that were faster (233 MHz) and had more capable graphics software packages with the latest version of OPENGL. Brief trouble was experienced in establishing the license server on the new UNIX machine, however, the local system administrator was able to solve the problem in a few days and AML was then functioning. If problems are encountered with UNIX AML set up or operation, find a UNIX system administrator who can help you and work closely with them and with the customer support at TSI to solve the problems. Persevere, and have patience.

Aeronautical Structural Analysis (ASTROS) Process

ASTROS Input File Process

Background: ASTROS is a comprehensive structural analysis software. The aeronautical analysis required for an aircraft model was performed by the Team using version 20.1 of the Automated Structural Optimization System (ASTROS) software. ASTROS resides on an SGI UNIX cluster of machines at the project sponsor, the Air Vehicles Directorate of the Air Force Research Laboratory, AFRL/VA, building 144, WPAFB. Under this sponsorship, we obtained an account from Mr. John Reakers in AFRL/VA on the IRIS Silicon Graphics, Incorporated (SGI) UNIX based machine.

The aircraft model was first constructed in AML and assigned a geometric mesh of quadrilateral (four sided) elements using the AML-MSC.PATRAN interface within AML. The node grid locations and element connectivity data was then formatted into an ASTROS input file (as text file with a ".d" suffix in the WordPad or NotePad PC text editors). This work was performed using PC computers at AFIT in the Systems Engineering room. A copy of the ASTROS manual was obtained from the sponsor and numerous examples of ASTROS input files were studied to gain relevant experience.

ASTROS Process:

The following steps outline the process of 1) Generating an ASTROS input file on a PC computer, 2) Transferring the file to a UNIX work station in such a manner that it becomes a UNIX readable file, and 3) Using the instructions in the input file to run the ASTROS optimization routine and perform the required battery of aeronautical analysis (including Static Analysis, Dynamic Analysis, Modal Analysis, etc.).

1) We first used a PC computer to generate a simple ASTROS input file (a text file with ".d" suffix) with the help of sponsor personnel. Further ASTROS and AML related questions may be directed to others in the same office with relevant experience such as Dr. Maxwell Blair, Ms. Victoria Tischler or Dr. Jeff Zweber. The ASTROS manual describing each section of the ASTROS input file was obtained from the sponsor also.

2) We then used the File Transfer Protocol WSFTP32 to transfer the ASTROS input file from the local PC computer in room 149C (the systems engineering room) at AFIT to the desired directory in our account on the IRIS UNIX work station at AFRL/VA. When transferring files from a DOS PC machine to a UNIX machine in WSFTP, one must check the ASCII file radio button instead of the default BINARY button. Otherwise, meaningless symbols are added to the end of each line of text which cause errors when the file is read by ASTROS. WSFTP was very useful in creating subdirectories in our UNIX account to organize the files we transferred.

3) To run ASTROS on the IRIS UNIX machine using the input file we had just transferred, a Telnet connection was required from the PC computers at AFIT to a UNIX command line on the IRIS machine. A PC computer which was connected to the AFIT network and, therefore, to the Internet was used for this purpose. The remote Telnet data connection was opened by clicking once on the "Start" bar in the bottom left corner of the PC computer screen (while running Windows 95). The "run" command was then clicked once and the word "command" was keyed in and the "okay" button was clicked once. An MS-DOS window was opened

and the command "telnet fibiris.flight.wpafb.af.mil" was entered where the word "telnet" is followed by the IP address of the IRIS UNIX machine. After entering the appropriate IRIS log on information, UNIX commands such as "ls" for "list contents of current directory" and "pwd" for "print the current working directory path" were used to open the path to our subdirectory where the newly transferred ASTROS input file resided.

4) The ASTROS software was run by entering the appropriate command at the UNIX command line on the Telnet connection to the IRIS machine at AFRL/VA. When the path to the proper directory is opened, ASTROS was executed using the instructions in the input file by entering the line "ASTROS *filename.d*". The "*filename.d*" represents the ASTROS input file with ".d" suffix generated previously. This command (barring any errors) then produced an ASTROS output file in the same directory with a ".prt" suffix. Many errors were generated by running input files. Each time, the ".prt" file was inspected for errors and the problems resolved in the ".d" input file.

Comments on the Process:

Much work was performed working with sponsor personnel to debug the ASTROS input files, ensure the proper ASTROS analyses were invoked and that the correct finite element constraints were imposed. Find a competent person with ASTROS experience and enlist their help. The ability to view the ASTROS model in a CAD type program such as HyperMesh or MSC.PATRAN is essential. The Team recommends acquiring an account on the Digital UNIX cluster of machines in the Aero department laboratory where MSC.PATRAN is resident.

ASTROS Input File Construction:

Most of the labor in composing an ASTROS input file lies in assigning the nodes their three-dimensional grid locations, then defining all finite elements in terms of the nodes that compose them. The Team was instructed by the sponsor to use quadrilateral elements known as CQUAD4 elements for the wing top and bottom skins and CSHEAR elements for all spars and ribs. See the examples in Appendix B under the "ASTROS code" section.

The ASTROS user directs the system through an input data stream composed of a command to attach the ASTROS run time database followed by multiple Data Packets. Each packet contains a set of related data providing the information needed to execute ASTROS. The packets begin with a keyword indicating the nature of the data within the packet and terminate with an ending keyword or with the start of the next data packet. files are composed of four sections: Assign Database, Solution, Begin Bulk and Connectivity sections

Solution Control Packet:

The solution control packet provides the means by which the user selects the optimization and analysis tasks to be performed by the ASTROS system, their order of execution and the engineering data related to each. The solution control command structure follows directly from the ASTROS capability to perform multidisciplinary analysis in a single run. The solution control packet is initiated with the keyword SOLUTION in the input data stream. The data are composed of solution control statements which can begin in any column and can extend over multiple physical records.

Each statement is formed from a combination of keywords separated by blank spaces or commas. Each command keyword can be abbreviated by the first four (or more) characters of the keyword. The solution control packet follows a prescribed hierarchy with the following levels:

Initial level (Level 1)
Type of Boundary Condition (Level 2)
Boundary Condition (Level 3)
Discipline (Level 4)

Further details on the use of the Optimize and Analyze conditions in ASTROS may be found in the ASTROS User's Manual.

APPENDIX C: SOFTWARE CODES AND EXPLANATIONS

C.1 Purpose

The intent of this appendix is to provide the software codes developed in this effort and documentation to allow future users to replicate and improve on the results in this document. The appendix contains the AML codes developed by the team and explanations of the code in a form similar to the AML reference manual.

C.2 AML Code Basics

As explained elsewhere in this document, AML is an object oriented modeling language. The AML coding language is based on LISP. Objects contain three categories of substructure: inheritances, properties, and subobjects. An object can inherit its data structure, type, and properties from other objects; properties can be defined within the object; objects can have subobjects attached to them, which are treated by AML as full objects. Thus, an AML model can have layers upon layers of nestled, specialized objects.

To create an instance of an object, the code defining the object must be loaded into memory. This can be done from the AML command line or the AML file editor. once loaded into memory, an object, when instantiated, will be compiled and created. As with most high-level computer languages, the successful loading or compiling of code does not mean that the object will be successfully instantiated.

C.3 AML File Management Basics

AML also allows for the formation of "systems," which are collections of files designed to be loaded and compiled together. This allows for a complex system of objects to be created from a collection of small-sized files. A system consists of a file folder containing a system.def file, which lists the component files existing in a subfolder, and the subfolder containing the source code. A compiled system also has a subfolder containing the compiled code. Depending on how they are coded, individual files in the system may have objects which can be instantiated without the complete system being loaded and compiled.

C.4 Design Team File Management

The codes created by the Design Team exist as a system, currently called "afit-airplane". The afit-airplane system currently contains 10 active files. The "parent" file is afit-airplane-object.aml, which contains object definitions which call on the remaining files of the system.

C.5 Supporting Add-Ons Required For afit-airplane

The current afit-airplane requires the "morphing" system to be loaded prior to loading the afit-airplane. If the afit-airplane system is run on a UNIX station, ensure that the AML-to-PATRAN interface system is loaded; if afit-airplane is run on a PC, ensure that the objects which use PATRAN are commented out. (These are noted as such in the next section.)

C.6 afit-airplane Object Explanation

afit-airplane-object **CLASS**

Inherits from: coordinate-system-class

Intent: afit-airplane-object is the parent object in the afit-airplane system. Design parameters are set from it, and configurations and aircraft components are its subobjects.

Comments: This object is not geometric in nature. It serves as the single point for changing parameters, and collects all subobjects associated with the afit-airplane system.

User Defined Properties:

aircraft-length	in units of length. Defines the distance from nosetip to tail.
aircraft-width	in units of length. Defines the distance from the centerline of the aircraft to the wingtip
fuselage-width	in units of length. Defines the distance from the centerline of the aircraft to the outer fuselage wall at the point where the leading edge of the wing intersects the fuselage
nose-angle	in degrees. Defines the angle made between the outside of the plane at the nose and the centerline of the plane
tail-angle	in degrees. Defines the angle made between the outside of the plane at the tail and the centerline of the plane
forward-fuselage-width-percent	the width of the fuselage, as a percent of the parameter "fuselage-width", at the point where the nose joins the fuselage
trailing-edge-fuselage-width-percent	fraction. the width of the fuselage, as a fraction of the parameter "fuselage-width", at the intersection of trailing edge of the wing and fuselage
aft-fuselage-width-percent	fraction. the width of the fuselage, as a fraction of the parameter "fuselage-width", at the point where the tail and fuselage meet.
station0-height-percent	fraction. height of the plane at station 0 (nose) as fraction of the calculated fuselage width at that point. For instance, a value of 0.5 results in the cross section of the fuselage being defined as an ellipse with a = 2b at station 0.
station1-height-percent	fraction. height of the plane at station 1 as fraction of the calculated fuselage width at that point.
station2-height-percent	fraction. height of the plane at station 2 as a fraction of the calculated fuselage width at that point.
station5-height-percent	fraction. height of the plane at station 5 as a fraction of the calculated fuselage width at that point.
station6-height-percent	fraction. height of the plane at station 6 as a fraction of the calculated fuselage width at that point.
station7-height-percent	fraction. height of the plane at station 7 as a fraction of the calculated fuselage width at that point.

sweep-angle	in degrees. deviation from perpendicular of the angle that the wing makes with the centerline of the aircraft. Positive values result in traditional delta wing; negative values create a swept forward wing.
wing-taper	fraction. Ratio of the chord length of the wing tip to chord length at the root of the wing
x-chord-length-at-root	in units of length. The chord length at the root of the wing, measured exclusively in the x-direction (parallel to the centerline of the aircraft).
leading-edge-location-percent	fraction. The point at which the leading edge of the wing intersects the fuselage, as a fraction of the parameter "aircraft-length"
profile	text. The NACA four digit profile of the wing.
wing-box-chord-front-percent	fraction. The point at which the front of the wing box begins, defined as a fraction of the chord length of the wing, with 0 defined as the leading edge of the wing.
wing-box-chord-back-percent	fraction. The point at which the rear of the wing box is, defined as a fraction of the chord length of the wing, with 0 defined as the leading edge of the wing.
rib-quantity	number of internal ribs to be created in the wing box
spar-quantity	number of internal spars to be created in the wing box
fuselage-wall-thickness-factor	fraction. Determines the diameter of the cargo area of the plane. Scales the cargo area diameters as a fraction of the fuselage diameters at the appropriate stations. A value of 0 results in no cargo area; a value of 1 results in the cargo area diameter equaling the fuselage diameter.
tag-attributes	text list. Exactly the same format as the AML property "tag-attributes" for any AML tagged-object. Defines the attributes used for tagging and meshing the model. The second parameter, minimum mesh size, is typically the only parameter changed.
tag-dimensions	list. Exactly the same format as the AML property "tag-dimensions" for any AML tagged-object. Controls which types of meshings are allowed for objects, with 2 denoting surface meshes, and 3 solid meshes.

Calculated-Properties:
NONE

Subobjects:

aircraft-union	A union-object of the fuselage subobject, right-wing subobject, and left-wing subobject
fuselage-skin	A body-morphing-object created from the children of the fuselage-sections subobject
wing-skin	An instance of afit-wing-ssc-object which contains the planform stations developed in the planform subobject

fuselage-sections	An instance of afit-fuselage-sections-object which contains the planform stations developed in the planform subobject
planform	An instance of afit-planform-outline-object
left-wing	A mirror-object of the right-wing
right-wing	A capped-surface-object of the previously mentioned wing-skin subobject
fuselage	A capped-surface-object of the previously mentioned fuselage-skin subobject
right-wing-box	An intersection-object of the super-wing-box-prism subobject (discussed below) and the right-wing subobject.
subgeom-wing-box	A sub-geom-object of the right-wing-box which generates six 2D surfaces from the right-wing-box
super-wing-box-prism	A capped-surface-object of the super-wing-box-skin object
super-wing-box-skin	An instance of afit-wing-box-ssc-object which contains the planform stations developed in the planform subobject
super-spar-sections	An instance of afit-spar-object which contains the planform stations developed in the planform subobject
super-angled-spar-sections	An instance of afit-angled-spar-object which contains the planform stations developed in the planform subobject
super-rib-sections	An instance of afit-rib-object which contains the planform stations developed in the planform subobject
series-spar	A series-object which has as its children the spars of the aircraft's wing.
series-angled-spar	A series-object which has as its children the spars of the aircraft's wing. The spars are angled so that none intersect the leading edge of the wing box
series-rib	A series-object which has as its children the ribs of the aircraft's wing
wing-box-union	A union-object of the ribs, spars, and outer skin of the wing box
cargo-hold	A body-morphing-object created from the children of the cargo-hold-sections subobject
cargo-hold-sections	An instance of afit-cargo-hold-sections-object which contains the planform stations developed in the planform subobject
cargo-objects	An instance of afit-cargo-object which contains the planform stations developed in the planform subobject

The subobjects of afit-airplane-object or the classes of them are defined below:

afit-angled-spar-object **CLASS**

Inherits-from: series-object

Intent: creates a series of large sheets aligned with each spar so that when this object is intersectioned with a wing box, the actual spars result. The sheets are placed

equidistantly between the leading edge and trailing edge of the wing box, as measured at the centerline of the aircraft. The sheets are oriented so that none of them intersect the leading or trailing edges of the wing box

Comments: Fundamentally the same as afit-spar-object, but with each sheet capable of getting a separate orientation. Relies on inputs sent via a planform-ptr pointer from the calling object

User Defined Properties:
Within the init-form property:

color	normal AML color selection property
render	normal AML rendering property (NOTE: the children of afit-rib-objects are normally not drawn.)

Calculated Properties:

spar-size	length. the maximum of three times the aircraft width or 1.5 times the aircraft length. Used by init-form for the length and width of the sheet. Purposely oversized.
spar-workspace	length. the distance between the front of the wing box and the rear of the wing box, at the aircraft's centerline.
quantity	integer. taken as the spar-quantity pointed to by planform-ptr, typically in afit-airplane-object.
outerstation-x	
innerstation-x	
outerstation-y	
innerstation-y	Defines the x and y locations of the inner- and outermost points on the leading edge of the wing box. Used within init-form to simplify the orientation expression.

Subobjects:
This series object consists of <quantity> (defined above) of sheet-objects.

afit-cargo-hold-sections-object **CLASS**

Inherits from: series-object

Intent: Currently, creates a smaller version of the fuselage to represent the cargo hold of the aircraft. Creates ellipses which are slightly smaller (based on the parameter "fuselage-wall-thickness-factor" pointed to by planform-ptr) than the fuselage.

Comments: Largely dependent on the definition of the fuselage cross-sections; code largely that of the fuselage-sections object.

User Defined Properties:

planform-ptr	The pointer by which the ellipse information is provided to the object

Calculated Properties:

width-list	a list of the width-radii of the fuselage at the stations where ellipses are to be generated
height-list	a list of the height-radii of the fuselage at the stations where ellipses are to be generated
orient-list	a list of the distances from the nose for the stations where ellipses are to be generated
a-dim	list. generates width diameters from the width radii in width-list
b-dim	list. generates height diameters from the height radii in height-list
quantity	number of ellipses to draw, based on the population of the width-list.

Subobjects:
The subsidiary objects are the ellipses themselves.

afit-cargo-object **CLASS**

Inherits from: coordinate-system-class

Intent: draws the rectangular cargo object, or series of objects, to be placed inside the aircraft.

Comments: The user can instantiate one of the subobjects to see how the cargo objects fit into the fuselage/plane. Dimensions of the rectangular cargo object are controlled by settings in this object, not the planform object. However, this object cannot be executed by itself, since it needs other information passed to it from the planform object

User Defined Properties:

planform-ptr	The pointer by which other geometric information is provided to the object
quantity	number of cargo objects created by series object
obj-length	length of object (along the major axis of the aircraft)
obj-width	width of object (along the transverse axis of the aircraft)
obj-height	height of the object
spacing	number. For a series, empty space between the cargo objects
x-offset	number. Space between the front of the fuselage (station 1) and the front of the cargo object

Calculated Properties:
NONE

Subobjects:

base-cargo-object	a single cargo-object, based on the user defined properties above. Instantiating this object will result in a single cargo-object, centered at the origin.
in-line-group	a series of cargo-objects which are placed one behind each other along the major axis of the aircraft (a la traditional aircraft loading)
lateral-group	a series of cargo-objects which are placed side to side along the transverse (wing) axis

afit-fuselage-sections-object **CLASS**

Inherits-from: series-object

Intent: create a series of elliptical cross-sections for the aircraft fuselage.

Comments: the output of this object is used to generate a skinned surface along the fuselage (which is then capped by another subobject). This object just contains the fuselage cross sections (ellipses). Relies on inputs sent via a planform-ptr pointer from the calling object

User Defined Properties:

planform-ptr	The pointer by which the ellipse information is provided to the object

Calculated Properties:

width-list	a list of the width-radii of the fuselage at the stations where ellipses are to be generated
height-list	a list of the height-radii of the fuselage at the stations where ellipses are to be generated
orient-list	a list of the distances from the nose for the stations where ellipses are to be generated
a-dim	list. generates width diameters from the width radii in width-list
b-dim	list. generates height diameters from the height radii in height-list
quantity	determines number of ellipses to draw, based on the population of the width-list.

Subobjects:
The subsidiary objects are the ellipses themselves.

afit-planform-outline-object **CLASS**

Inherits from: polygon-object

Intent: creates a planform outline of the aircraft. Also calculates the station locations of the aircraft. The station locations are used as the vertices of the polygon-object

Comments: station locations are used by just about every other class associated with afit-airplane system. All "z" locations are currently hard coded as 0.0

User Defined Properties:
NONE

Calculated Properties:
This object's code contains a list of the aircraft parameters previously defined in afit-airplane-object. The values for these parameters are taken from afit-airplane-object when this object is called from it.

station-0x station-0y station-0z	The location of station 0: nose tip
station-1x station-1y station-1z	The location of station 1: nose ends, fuselage begins
station-2x station-2y station-2z	The location of station 2: leading edge of wing intersects fuselage
station-2cx station-2cy station-2cz	The location of station 2c: leading edge of wing extended to meet centerline of aircraft
station-3x station-3y station-3z	The location of station 3: leading edge of wing at wing tip
station-4x station-4y station-4z	The location of station 4: trailing edge of wing at wing tip
station-5x station-5y station-5z	The location of station 5: trailing edge of wing intersects fuselage
station-6x station-6y station-6z	The location of station 6: fuselage ends, tailcone begins
station-7x station-7y station-7z	The location of station 7: tail ends

Subobjects:
NONE

afit-rib-object **CLASS**

Inherits from: series-object

Intent: creates a series of large sheets aligned with each rib so that when this object is intersectioned with a wing box, the actual ribs result. The sheets are placed equidistantly between the centerline and the tip of the wing.

Comments: the sheets are purposely oversized so that no matter the shape, location, or orientation of the wing, ribs can be created. Relies on inputs sent via a planform-ptr pointer from the calling object

User Defined Properties:

y-axis-translation	length. in case the centerline is not located along the x=0 line, moves the rib-workspace along the y-axis a given distance

Within the init-form property:

color	normal AML color selection property
render	normal AML rendering property (NOTE: the children of afit-rib-objects are normally not drawn.)

Calculated Properties:

rib-size	length. the maximum of twice the aircraft width or 1.5 times the aircraft length. Used by init-form for the length and width of the sheet. Purposely oversized.
rib-workspace	length. the distance between the point along the tip of the wing closest to the centerline and the centerline of the aircraft. Sets the space in which the ribs are to be placed.
quantity	integer. taken as the rib-quantity pointed to by planform-ptr, typically in afit-airplane-object.

Subobjects:
This series object consists of <quantity> (defined above) of sheet-objects.

afit-spar-object **CLASS**

Inherits-from: series-object

Intent: creates a series of large sheets aligned with each spar so that when this object is intersectioned with a wing box, the actual spars result. The sheets are placed equidistantly between the leading edge and trailing edge of the wing box, as measured at the centerline of the aircraft.

Comments: the sheets are purposely oversized so that no matter the shape, location, or orientation of the wing, spars can be created. Relies on inputs sent via a planform-ptr pointer from the calling object

User Defined Properties:
Within the init-form property:

color	normal AML color selection property
render	normal AML rendering property (NOTE: the children of afit-rib-objects are normally not drawn.)

Calculated Properties:

spar-size	length. the maximum of three times the aircraft width or 1.5 times the aircraft length. Used by init-form for the length and width of the sheet. Purposely oversized.
spar-workspace	length. the distance between the front of the wing box and the rear of the wing box, at the aircraft's centerline.
spar-angle	degrees. sets the orientation angle of the sheets equal to the angle which is made by the trailing edge of the wing box.
quantity	integer. taken as the spar-quantity pointed to by planform-ptr, typically in afit-airplane-object.

Subobjects:
This series object consists of <quantity> (defined above) of sheet-objects.

afit-wing-box-ssc-object **CLASS**

Inherits from: skin-surface-from-curves object

Intent: Creates a skinned surface (four-sided) from two polygons: one defined at the centerline of the aircraft and one at the wing tip. The skinned surface is the parent of one of the objects which is later intersected to create the wing box.

Comments: The sides of the polygons represent the leading and trailing edge of the wing box, while the top and bottom are purposely drawn at a large distance from the z=0 plane so that any wing shape that is intersected with them creates a full wing box.

User Defined Properties:

planform-ptr	The pointer by which the station location information is provided to the object
render	normal AML rendering property
web-surfaces?	true/nil. Default is NIL. Web-surfaces? creates a webbed, discontinuous surface when set to TRUE, otherwise it creates a nurbed, continuous surface when set to false.

Calculated Properties:

curve-objects	list of objects on which to base the skin. Default is centerline-section and tip-section, the two polygons defined in this object
centerline1x centerline1y centerline1z centerline2x centerline2y centerline2z centerline3x centerline3y centerline3z centerline4x centerline4y centerline4z tip1x tip1y tip1z tip2x tip2y tip2z tip3x tip3y tip3z tip4x tip4y tip4z	These are the x, y, and z locations for the four points of each of the two polygons. They are defined by the station locations and the wing box properties pointed to by planform-ptr

Subobjects:

centerline-section	a rectangle created along the aircraft centerline whose left and right sides are those of the wing-box
tip-section	a rectangle created along the wingtip whose left and right sides are those of the wing-box

afit-wing-ssc-object **CLASS**

Inherits from: skin-surface-from-curves-object

Intent: creates a wing as a skinned surface object, based on three airfoils created from information included in the planform property. afit-airplane-object uses the object created by this class as the basis for the right-wing.

Comments: The afit-wing-ssc-object was recreated after an afit-wing-object based on body-morphing-object did not work. If the outer points of the three airfoils can be

connected by straight lines, afit-wing-ssc-object will result in a wing defined by straight lines.

User Defined Properties:

planform-ptr	list. Default is nil; included so that when the object is called from afit-airplane-object there will be a location for information from the planform
render	normal AML render property
web-surfaces?	true/nil. Default is TRUE. Web-surfaces? creates a webbed, discontinuous surface when set to TRUE, otherwise it creates a nurbed, continuous surface when set to false.

Calculated Properties:

curve-objects	normal AML curve-objects property for a skin-surface-from-curves object. Default is the list of centerline-section, root-section, and tip-section, which are subobjects of afit-wing-ssc-object

Subobjects:

centerline-section	creates a NACA four digit airfoil of the type specified in planform-ptr (normally pointing to the planform subobject of afit-airplane-object) along the centerline of the aircraft. Placement is defined by the station locations which are pointed to by planform-ptr.
root-section	creates a NACA four digit airfoil of the type specified in planform-ptr at the junction of the wing and the fuselage, defined by the station locations which are pointed to by planform-ptr
tip-section	creates a NACA four digit airfoil of the type specified in planform-ptr at the tip of the wing, defined by the station locations which are pointed to by planform-ptr

aircraft-union **SUBOBJECT**

Object: afit-airplane-object

Instance Of: union-object

Intent: unions subobjects into a single entity. The union is not currently tagged.

User Defined Properties:
NONE

Calculated Properties:

object-list	list. List of subobjects to be joined. Default is fuselage, left-wing, and right-wing subobjects of the afit-airplane-object object.

cargo-hold **SUBOBJECT**

Instance of: body-morphing-object

Intent: creates the volume in which the cargo-objects are supposed to be contained

User Defined Properties:

render	The normal AML render property
web-surfaces?	true/nil. Default is NIL. Web-surfaces? creates a webbed, discontinuous surface when set to TRUE, otherwise it creates a nurbed, continuous surface when set to false.

Calculated Properties:

curves-to-morph	points to the children of the cargo-hold-sections

cargo-hold-sections **SUBOBJECT**

Instance of: afit-cargo-hold-sections-object

Intent: creates the cross-sections (ellipses) upon which the cargo-hold is defined

User Defined Properties:
NONE

Calculated Properties:

planform-ptr	should remain as planform object

cargo-objects **SUBOBJECT**

Instance of: afit-cargo-object

Intent: create the cargo objects for volumetric analysis of the aircraft

User Defined Properties:
NONE

Calculated Properties:

planform-ptr	should remain as planform object

fuselage **SUBOBJECT**

Instance of: capped-surface-object

Intent: creates the fuselage by capping the fuselage-skin object, creating a solid, completely bounded, object.

Comments: this object should be used to represent the fuselage

User Defined Properties:

render	The normal AML render property
color	The normal AML color property
solid?	default is "T". The "true" setting allows the Boolean objects which inherit from it to have volume

Calculated Properties:

source-object	should be left as wing-skin

fuselage-sections **SUBOBJECT**

Instance of: afit-fuselage-sections-object

Intent: Creates the cross-sectional ellipses that define the fuselage

Comment: drawing the object (rather, its children) draws the fuselage cross-sectional ellipses

User Defined Properties:

planform-ptr	points to the planform object. Should normally be left as is.

Calculated Properties:
NONE

fuselage-skin **SUBOBJECT**

Instance of: body-morphing-object

Intent: Creates the uncapped, skinned shell of the fuselage

Comments: normally, should not be drawn. Draw fuselage instead.

User Defined Properties:

render	normal AML rendering property

web-surfaces?	true/nil. Default is NIL. Web-surfaces? creates a webbed, discontinuous surface when set to TRUE, otherwise it creates a nurbed, continuous surface when set to false.

Calculated Properties:

curves-to-morph	points to the children of the fuselage-sections object

left-wing **SUBOBJECT**

Instance of: mirror-object

Intent: creates the left wing

Comments: the wing is drawn as a mirror of the right wing, so no changes as such should be done to this object

User Defined Properties:

render	The normal AML render property
color	The normal AML color property

Calculated Properties:

source-object	should be left as right-wing
basis-vector	Used by AML to generate the mirror object
point-on-mirror	Used by AML to generate the mirror object

planform **SUBOBJECT**

Instance of: afit-planform-outline-object

Intent: draws outline of aircraft, and makes geometric parameters available to other subobjects

Comments: Drawing this object creates the planform outline of the aircraft

User Defined Properties:
NONE

Calculated Properties:
NONE

right-wing **SUBOBJECT**

Instance of: capped-surface-object

Intent: creates the right wing by capping the wing-skin object to create a wing with "six" sides to it

Comments: this object should be used to generate pictures of the wing

User Defined Properties:

render	The normal AML render property
color	The normal AML color property
solid?	default is "T". The "true" setting allows the Boolean objects which inherit from right-wing to have volume

Calculated Properties:

source-object	should be left as wing-skin

right-wing-box **SUBOBJECT**

Instance of: intersection-object

Intent: creates the outside of wing-box by taking the common areas of super-wing-box-prism and right-wing

Comments: ensure that the objects which this object inherits from are solid objects

User Defined Properties:

render	The normal AML render property
color	The normal AML color property

Calculated Properties:

object-list	Should be left as a list containing super-wing-box-prism and right-wing

series-angled-spar **SUBOBJECT**

Instance of: series-object

Intent: creates the spars in the wing box as a series of intersection-objects

Comments: Uses a tagged-intersection-object to allow meshing of the object

User Defined Properties:

render	The normal AML render property
color	The normal AML color property
series-prefix	Default is "spar"

Calculated Properties

quantity	set to refer to spar-quantity
class-expression	should be left as tagged-intersection-object to allow for meshing
spar-list	should be left as the children of the super-angled-spar-sections
init-form	contains the additional properties tag-attributes and tag-dimensions, which point to the global values for these properties

series-rib **SUBOBJECT**

Instance of: series-object

Intent: creates the ribs in the wing box as a series of intersection-objects

Comments: Uses a tagged-intersection-object to allow meshing of the object

User Defined Properties:

render	The normal AML render property
color	The normal AML color property
series-prefix	Default is "rib"

Calculated Properties

quantity	set to refer to rib-quantity
class-expression	should be left as tagged-intersection-object to allow for meshing
spar-list	should be left as the children of the super-rib-sections
init-form	contains the additional properties tag-attributes and tag-dimensions, which point to the global values for these properties

series-spar **SUBOBJECT**

Instance of: series-object

Intent: creates the spars in the wing box as a series of intersection-objects

Comments: Uses a tagged-intersection-object to allow meshing of the object

User Defined Properties:

render	The normal AML render property
color	The normal AML color property
series-prefix	Default is "spar"

Calculated Properties

quantity	set to refer to spar-quantity
class-expression	should be left as tagged-intersection-object to allow for meshing
spar-list	should be left as the children of the super-spar-sections
init-form	contains the additional properties tag-attributes and tag-dimensions, which point to the global values for these properties

sub-geom-wing-box **SUBOBJECT**

Instance of: sub-geom-object

Intent: rearranges the right-wing-box to allow each surface to be its own object

Comment: needed so that the individual surfaces of the outside of the wing box can be treated as the ribs and spars are, and can be intersected and unioned.

User Defined Properties:
NONE

Calculated Properties:

source-object	Should be left as right-wing-box
sub-geom-dimension	Should be left as 2 (which greps the 2-D objects (surfaces) from the source-object

super-angled-spar-sections **SUBOBJECT**

Instance of: afit-angled-spar-object

Intent: create the series of sheets oriented as the spars should be

Comments: This object creates large sized sheets, which are intersected to give the proper geometry. See the afit-angled-spar-object and afit-spar-object about the difference between the two types of spars

User Defined Properties:
NONE

Calculated Properties:

planform-ptr	Should be left as the planform object

super-rib-sections **SUBOBJECT**

Instance of: afit-rib-object

Intent: create the series of sheets oriented as the ribs should be

Comments: This object creates large sized sheets, which are intersected to give the proper geometry.

User Defined Properties:
NONE

Calculated Properties:

planform-ptr	Should be left as the planform object

super-spar-sections **SUBOBJECT**

Instance of: afit-spar-object

Intent: create the series of sheets oriented as the spars should be

Comments: This object creates large sized sheets, which are intersected to give the proper geometry.

User Defined Properties:
NONE

Calculated Properties:

planform-ptr	Should be left as the planform object

super-wing-box-prism **SUBOBJECT**

Instance of: capped-surface-object

Intent: creates the solid surface to be intersected with the wing to form the wing-box

User Defined Properties:

render	The normal AML render property
color	The normal AML color property
solid?	default is "T". The "true" setting allows the Boolean objects which inherit from it to have volume

Calculated Properties:

source-object	should be left as super-wing-box-skin

super-wing-box-skin **SUBOBJECT**

Instance of: afit-wing-box-ssc-object

Intent: Creates the uncapped, skinned shell of the object to be intersected with the wing to form the wing-box

Comments: Normally should not be drawn. Draw right-wing-box or super-wing-box-prism instead

User Defined Properties:
NONE

Calculated Properties:

planform-ptr	Should be left as the planform object

wing-skin **SUBOBJECT**

Instance of: afit-wing-ssc-object

Intent: Creates the uncapped, skinned shell of the wing

Comments: normally, should not be drawn. Draw right-wing or left-wing instead.

User Defined Properties:

planform-ptr	points to the planform object. Should normally be left as is.

Calculated Properties:
NONE

wing-box-union **SUBOBJECT**

Instance of: union-object

Intent: combines the components of the whole wing box into a single object

Comments: the difference between a series object and subgeom object may cause problems when this object is created.

User Defined Properties:

simplify?	default is "nil". When set to nil, the intersections of spars and ribs will be represented as a surface boundary (needed for meshing to

work properly). When set to "T", no surface boundary exists across an intersection

Calculated Properties:

object-list	should be left as the children of series-spar, series-rib, and subgeom-wing-box. series-angled-spar can be substituted for series-spar when desired.

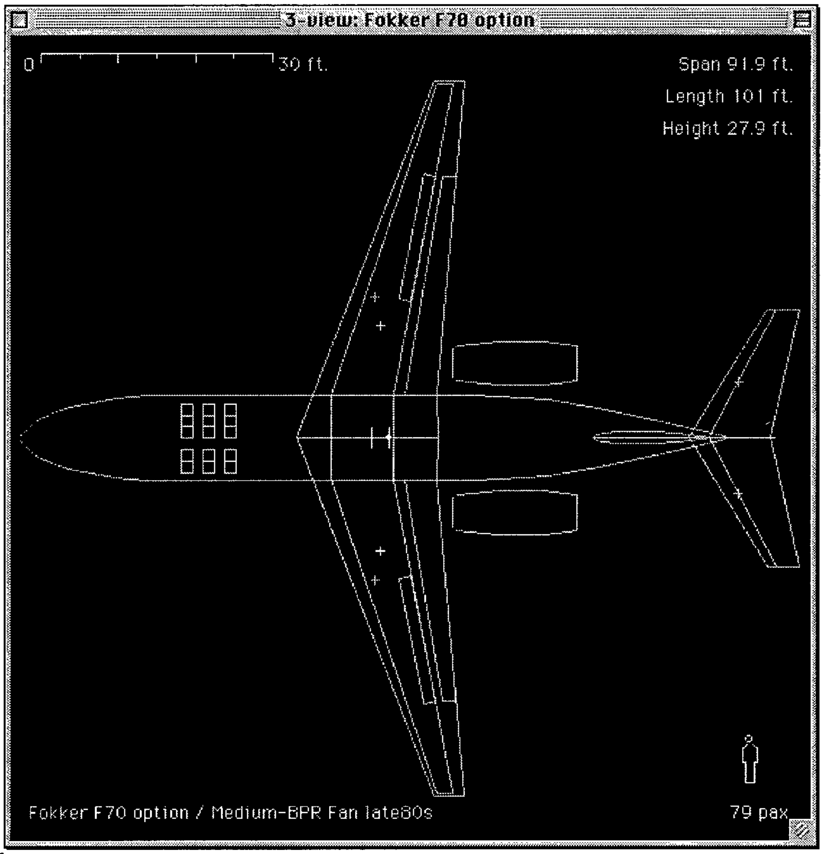
APPENDIX D: PIANO SOFTWARE AIRCRAFT CONCEPTUAL DESIGN EXAMPLE

Current Aircraft Conceptual Design Using PIANO Software.

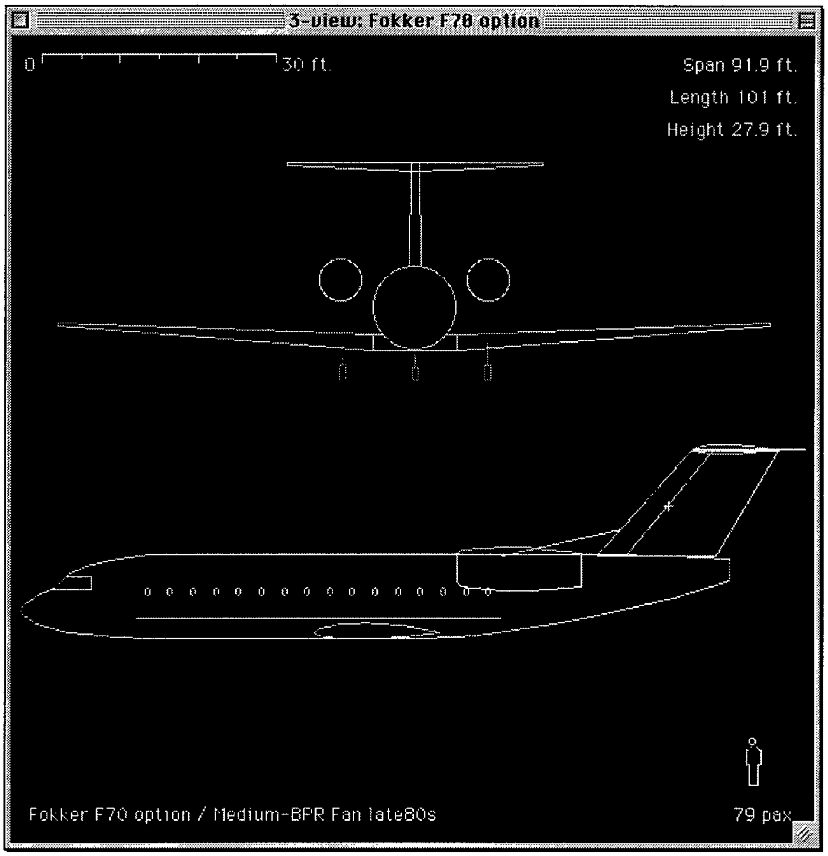
The following sample outputs were produced directly from the PIANO conceptual model of the medium commercial transport, Fokker 70. The results are known to match the manufacturer's claims quite well in areas where data are available. This is an independent analysis and does not necessarily reflect the manufacturer's formal position (Simos, 1998).

PIANO requires inputs for the Fokker F70 aircraft such as payload, maximum takeoff weight, and engine thrust. Given these and other inputs, PIANO evaluates all aspects of a conceptual aircraft design. Below are included the results of the PIANO aircraft evaluation. The full output included numerous measures of the aircraft geometry and design limits such as range, drag, emissions of NO_X pollutants, planned climb patterns, takeoff and landing performance and detailed price break downs including operations and support costs. The full results were too voluminous to include here and may be viewed along with additional information about PIANO at the PIANO web page address: http://www.lissys.demon.co.uk

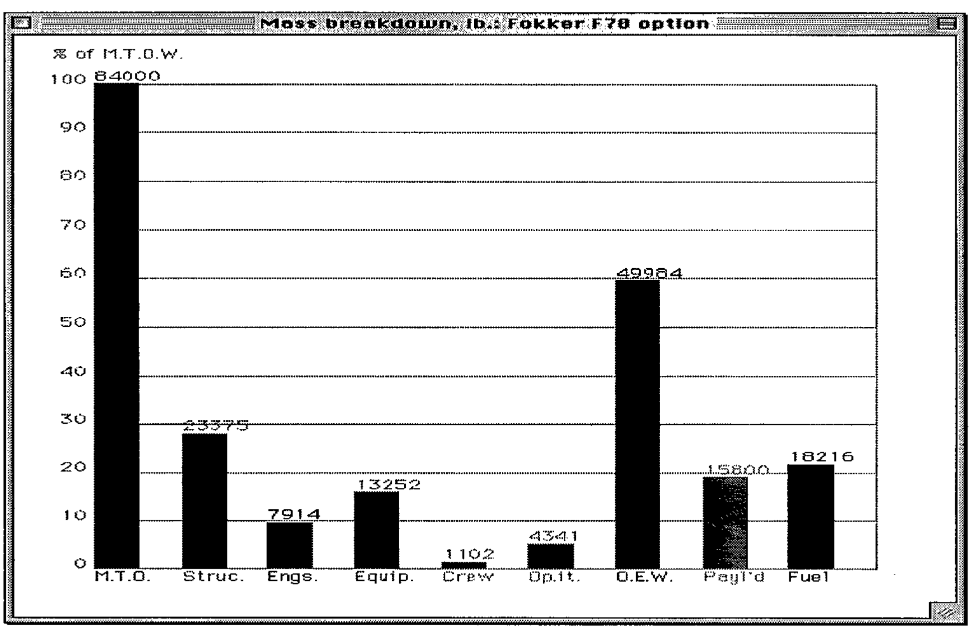
Below are sample graphical and pictorial output compiled directly from the Piano model of the Fokker F70. PIANO is a fine example of the software computer aided engineering tools available to aid in aircraft conceptual design.

- Plan View 
3-view: Fokker F70 option
0
30 ft.
Span 91.9 ft.
Length 101 ft.
Height 27.9 ft.
Fokker F70 option / Medium-BPR Fan late80s
79 pax

* Side and Front Views

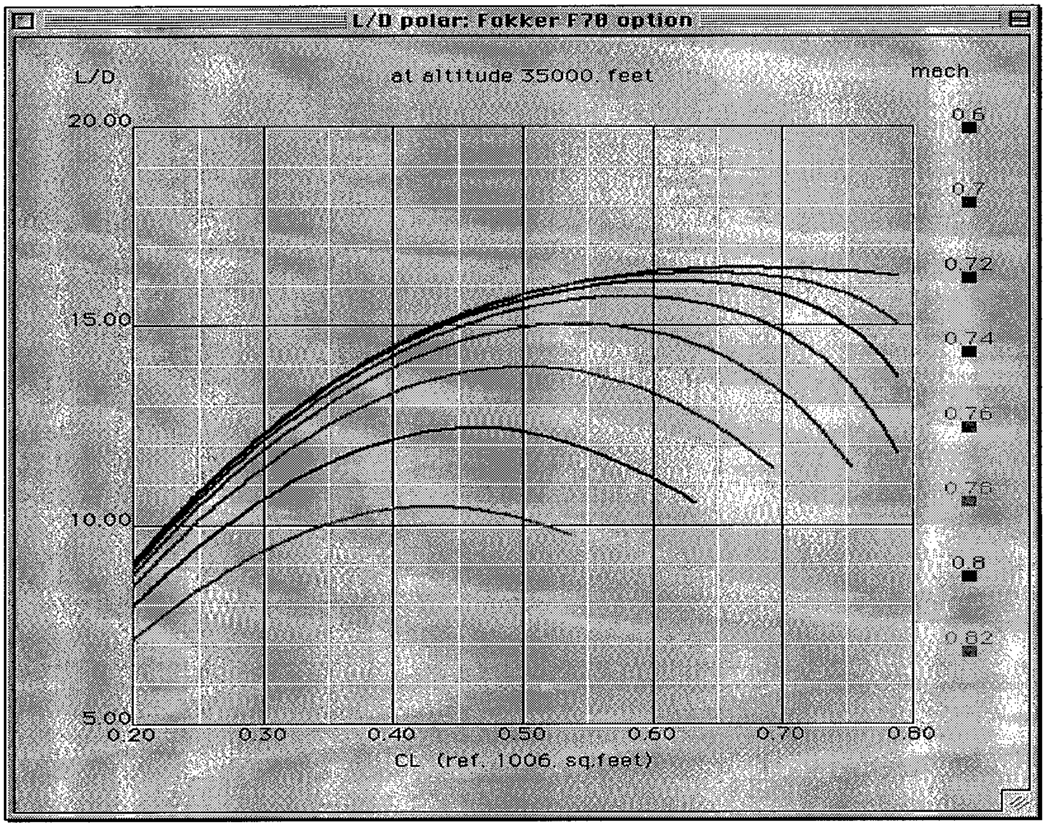


* Mass Barchart

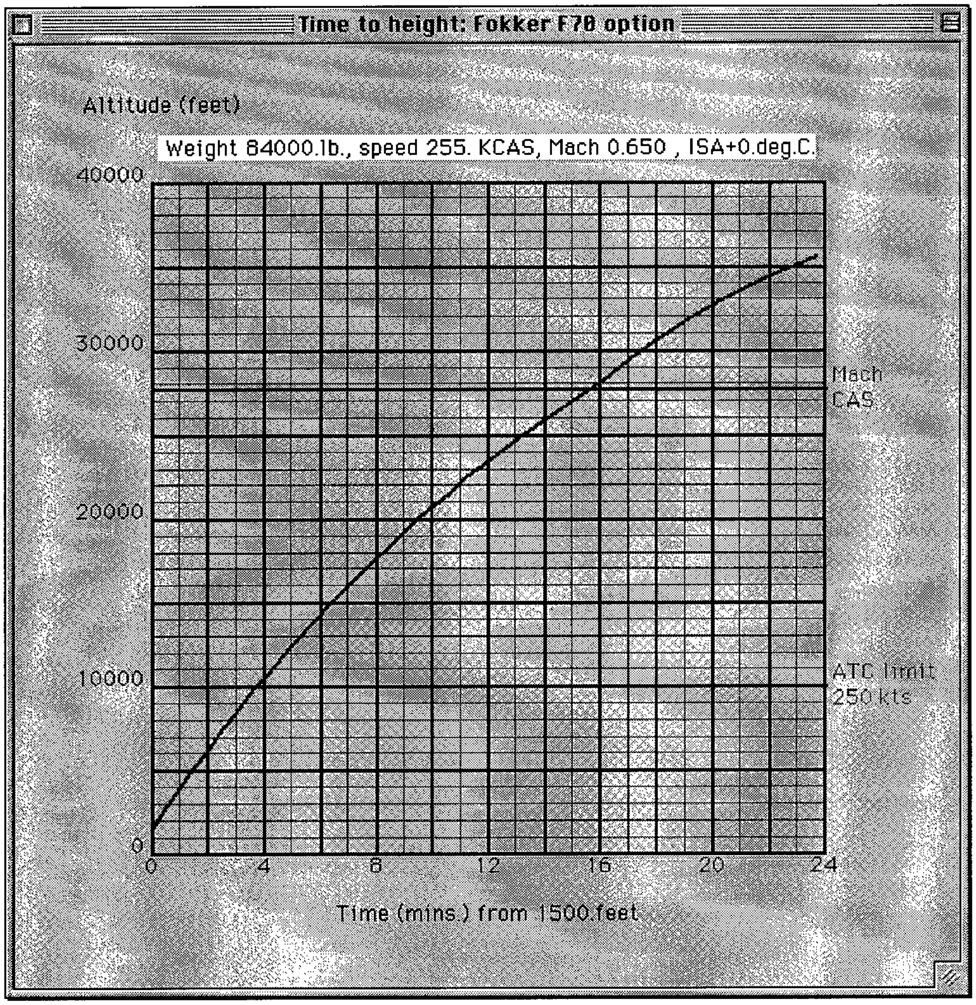


*

- Lift-Drag Polar



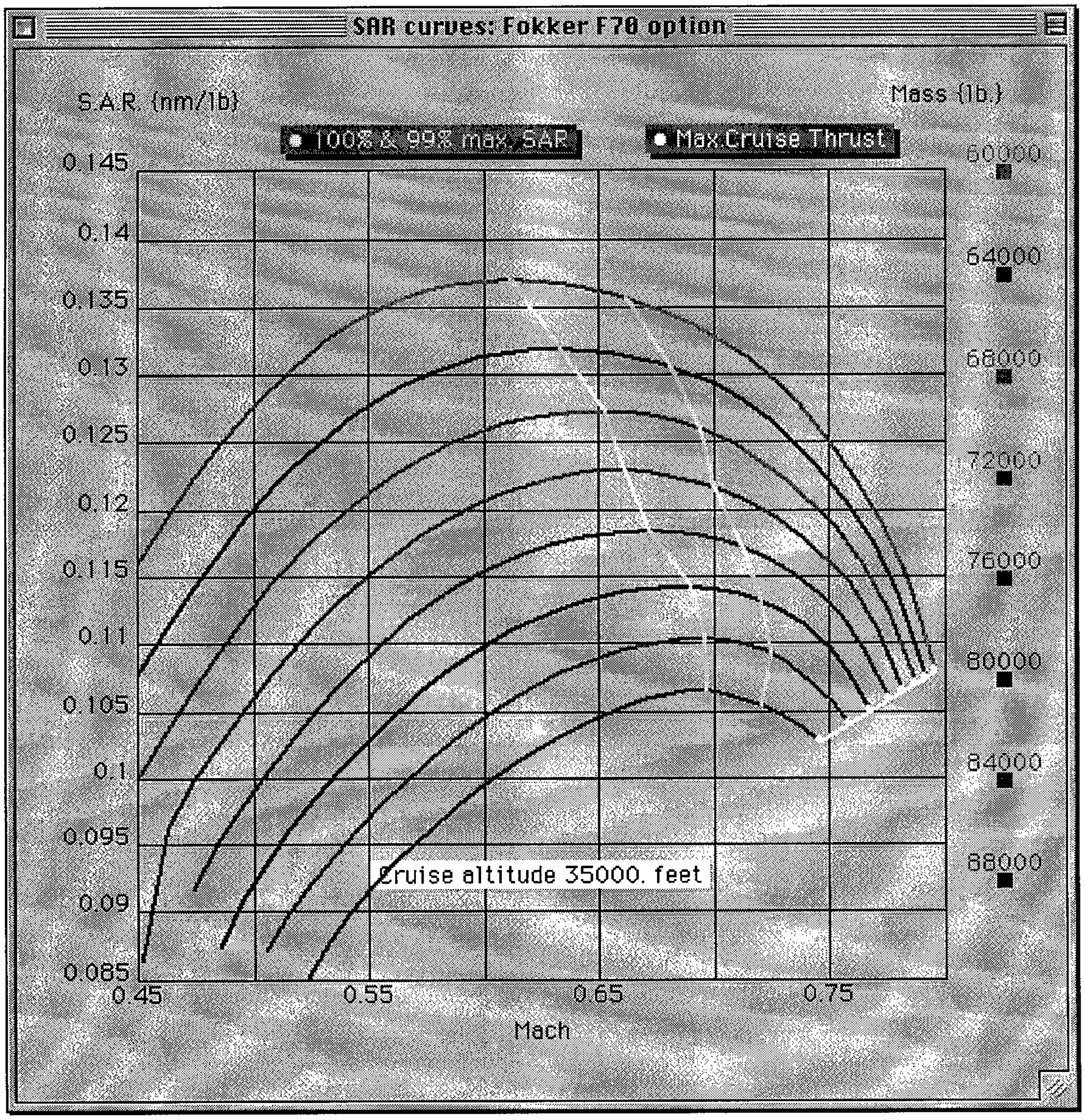
- Climb Time



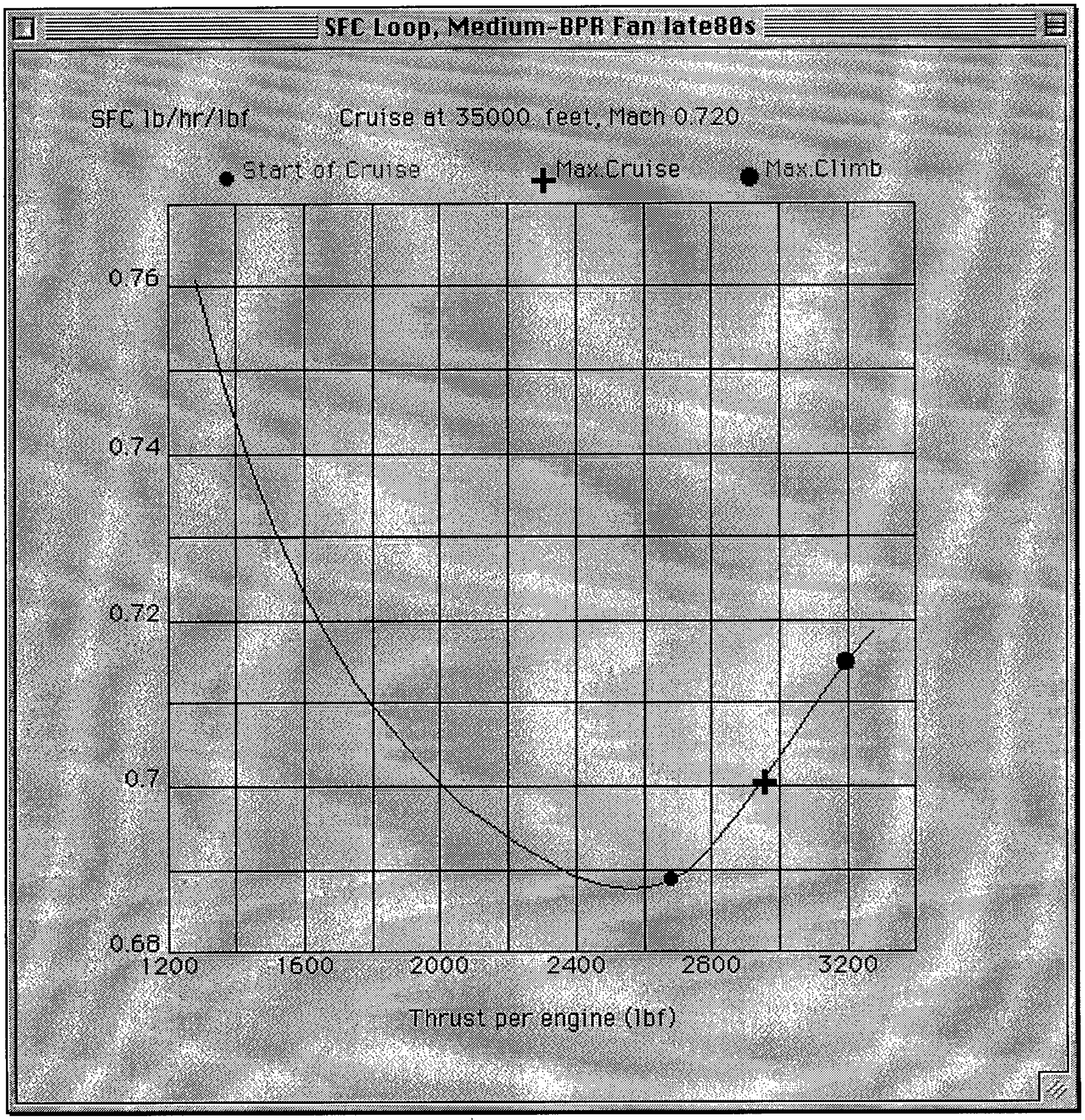
Time to height: Fokker F70 option
Altitude (feet)
Weight 84000.lb., speed 255. KCAS, Mach 0.650 , ISA+0.deg.C.
40000
30000
20000
10000
0
0
4
8
12
16
20
24
Mach
CAS
ATC limit
250 kts
Time (mins.) from 1500.feet

-

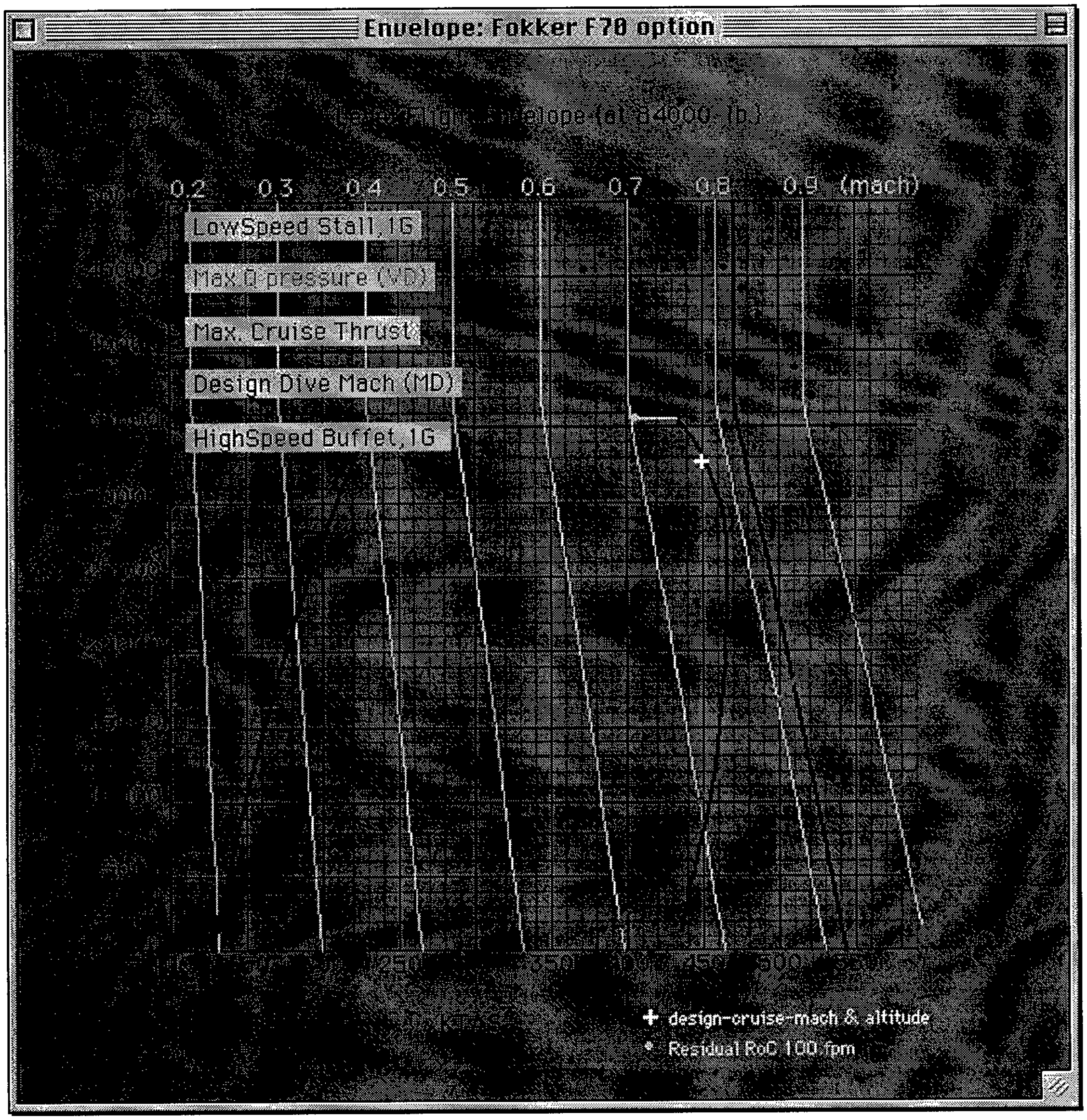
- Specific Air Range (NAMS)



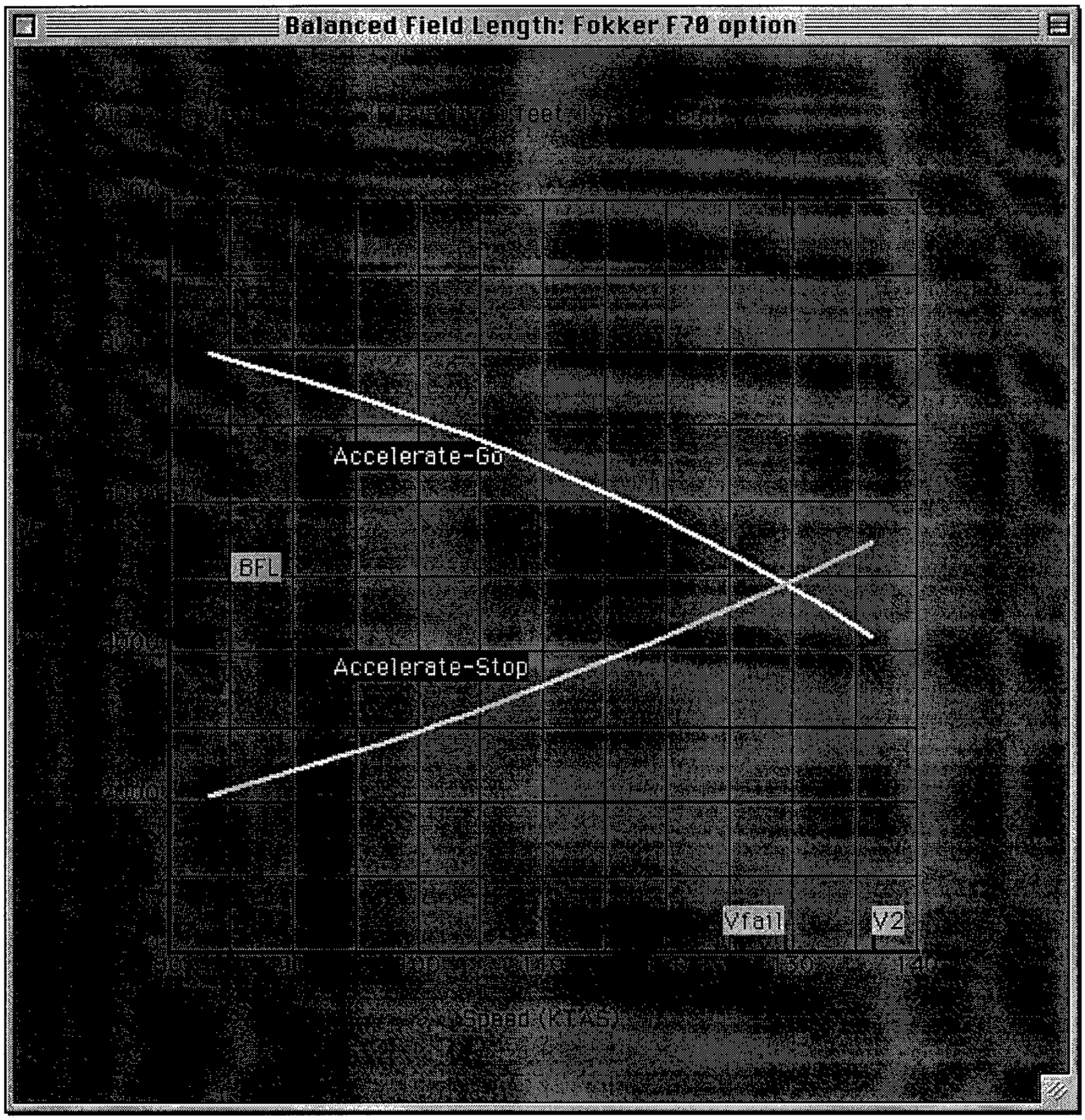
- Specific Fuel Consumption Cruise Loop



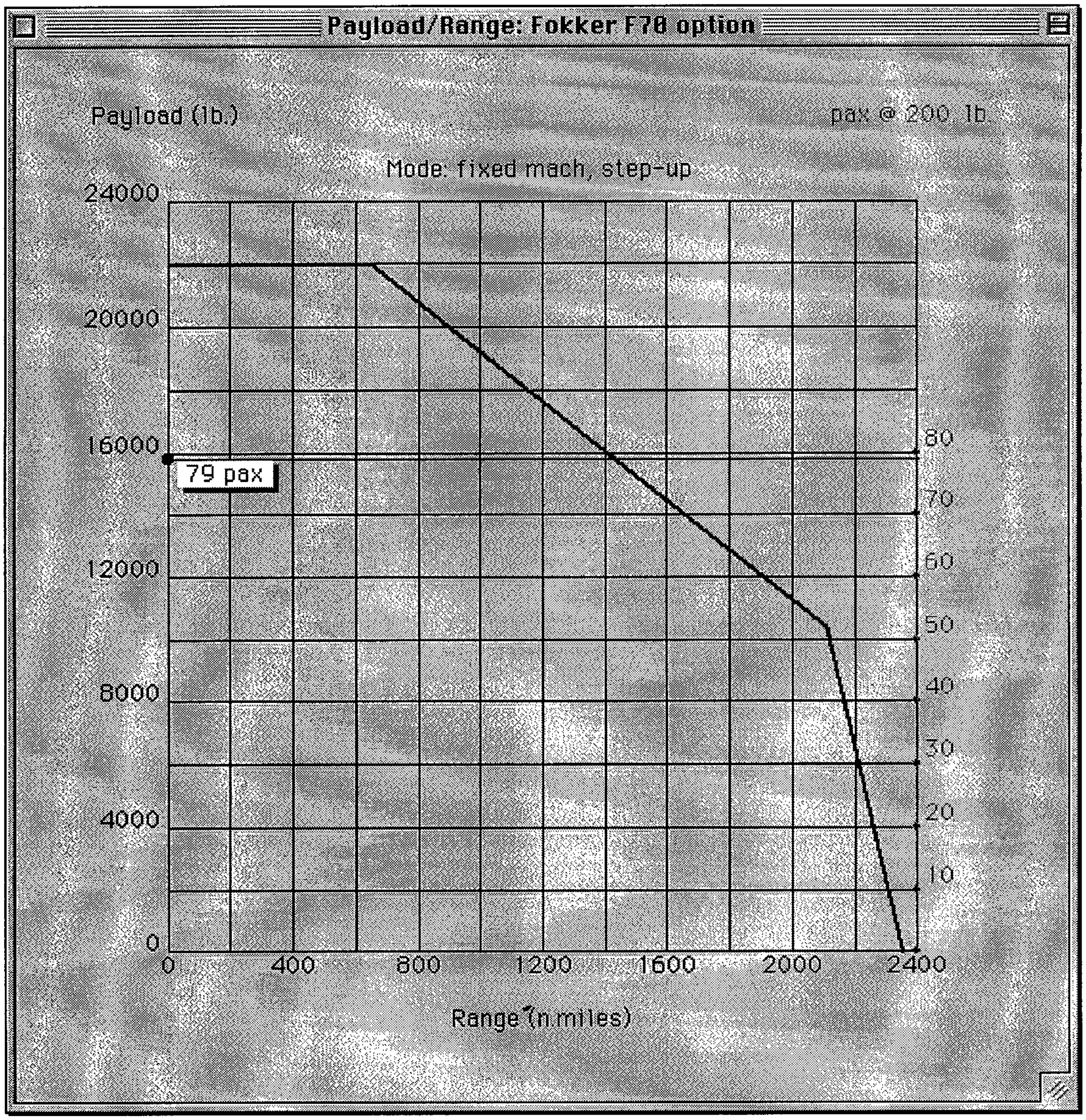
• Flight Envelope



- Balanced Field Length Calculation



- Payload-Range Diagram



Both existing and projected aircraft are modeled through basic parameters that can be assigned interactively, in any order. A full re-design procedure is executed automatically whenever a value is changed and if new output is requested. The system makes a number of consistency checks. On-line help is provided for all interactive features. More than 200 parameters are available, but most aircraft definitions typically

require only 50 to 60 of these. An indefinite number of aircraft ('point designs') can be generated and automatically saved in compact files.

There are three basic types of parameter in Piano:

- 'Vital' (red) parameters (such as aspect-ratio) which need to be initially supplied. They constitute the basic level of the definition. There are only about 20 of these.
- 'Defaulted' (black) parameters (such as mass-per-pax). These have a typical value which is used unless it is overridden by the user.
- 'Calculable' (green) parameters (such as APU-mass). Built-in estimation methods are used for these (as opposed to a simple fixed value). An alternative setting can be supplied by the user.

The following is a list of these parameters. It is provided here without detailed explanations, though the meaning of most should be fairly obvious:

aerofoil-clmax
aerofoil-cm0
air-condition-mass
airframe-$/mass
airframe-price-$
amortization-years
approach-method
approach-time
apu-mass
aspect-ratio
avionics-mass
blades-per-propeller
braking-friction
buffet-cl-adjustment
buffet-mach-adjustment
bypass-ratio
cabin-aisle-width
cabin-altitude
cabin-crew-$/hr
cabin-floor-location
cabin-in-front-fuse-fraction
cabin-in-rear-fuse-fraction

cabin-is-pressurised
cabin-seat-pitch
cabin-seat-width
cargo-doors-area
cd0-compress.start-mach
cd0-compressibility-factor
centresection-is-wet
climb-schedule-switch-alt.
compressibility-method
contingency-definition
contingency-fuel-fraction
cost-method
delta-cd-due-to-u/c
delta-clmax-due-to-slat
design-cruise-altitude
design-cruise-mach
design-dive-mach
design-floor-loading
design-n-lim
dihedral-deg
diversion-altitude-limit
diversion-distance
diversion-mach
dorsal-fin-height-fraction
dorsal-fin-length-fraction
drag-creep-slope
drag-creep-start
electric-systems-mass-fraction
engine-$/thrust
engine-pressure-ratio
engine-price-$
engine-type
eta-flap
eta-planform-break
eta-thickness-break
eta-u/c
exist-2nd-deck
exist-slats
exist-winglets
fairing-type
fin-aspect-ratio
fin-sweep-deg
fin-t/c
fin-tailcone-gap
fin-taper
fixed-equipment-cg-fraction

flap-chord-fraction
flap-type
flight-crew-$/hr
front-fuse-length
front-fuse-name
fuel-density
fuel-price-$/vol
fuel-systems-mass
fuel-vol-adjustment
furnishings-mass-per-pax
fuse-depth
fuse-mass-method
fuse-transition
fuse-width
fuse-xsection-type
hold-altitude
hold-mach
hold-time-mins
hydraulic-systems-mass-fraction
ignore-fuel-vol-violations
ignore-seating-checks
incidence-correction
interest-rate
labor-$/hr
landing-flap-deg
landing-mass/mto-mass
landing-screen-height
linked-engine-name
main-u/c-wheels-per-a/c
mass-per-crew
mass-per-pax
max-operating-altitude
max-payload/design-payload
mid-fuse-length
min-static-margin
misc-systems-mass-fraction
mto-mass
nac-depth
nac-length
nac-location-ahead-of-wing
nac-location-below-wing
nac-location-on-fuse
nac-mounted-on-fin
nac-name
nac-width
nac-depth

nac-length
nac-longitudinal-location
nac-name
nac-vertical-location
nac-width
nacs-mounted-on-fuse
nacs-mounted-on-wing
nose-u/c-wheels-per-a/c
number-of-cabin-crew
number-of-compressor-stages
number-of-flight-crew
number-of-pax
number-of-shafts
number-of-windows
operational-items-mass
pax-doors-area
planform-break-is-wet
planform-break-t.e.-adjustment
polar-mod-name
powerplant-thrust/weight
propeller-diameter
rear-fuse-length
rear-fuse-name
required-fin-vol-coeff
required-stab-vol-coeff
residual-value-fraction
reverse-thrust-fraction
reverse-thrust-used-for-landing
rolling-friction
roof-top-end
seats-abreast
skin-friction-method
sl-isa-static-thrust-per-engine
slat-chord-fraction
slat-exp-span-fraction
spoiler-chord-fraction
spoiler-exp-span-fraction
stab-aspect-ratio
stab-mounting
stab-sweep-deg
stab-t/c
stab-tailcone-gap
stab-taper
sweep-deg
t/c-break/root
t/c-root

t/c-tip/root
tail-mass-method
takeoff-flap-deg
takeoff-rotation-check
takeoff-screen-height
takeoff-time
taper
taxi-in-time
taxi-out-time
thrust-factor-at-2nd-segment
twist-deg
u/c-ground-clearance-fraction
u/c-mounted-on
user-cds-increment
user-factor-on-box-mass
user-factor-on-climb-rating
user-factor-on-continuous-rating
user-factor-on-cruise-rating
user-factor-on-divergence-mach
user-factor-on-fin-drag
user-factor-on-fin-mass
user-factor-on-flap-mass
user-factor-on-fuse-drag
user-factor-on-fuse-mass
user-factor-on-induced-drag
user-factor-on-landing-clmax
user-factor-on-landing-l/d
user-factor-on-nac-drag
user-factor-on-sfc
user-factor-on-stab-drag
user-factor-on-stab-mass
user-factor-on-takeoff-clmax
user-factor-on-takeoff-l/d
user-factor-on-takeoff-rating
user-factor-on-u/c-mass
user-factor-on-wing-drag
v2-speed-ratio
window-depth
window-width
windscreen-depth
windscreen-frontal-cd
windscreen-top-fraction
windscreen-width-fraction
wing-apex-fuse-fraction
wing-area
wing-mass-method

wing-mounting
wing-transition
winglet-cant-deg
winglet-root-chord/wing-tip-chord
winglet-span/wing-halfspan
xi-front-spar-root
xi-front-spar-tip
xi-rear-spar-root
xi-rear-spar-tip
xi-sweep

BIBLIOGRAPHY

Adaptive Modeling Language [Version 3.1.3] Reference Manual. Electronic publication, bundled with computer software. TechnoSoft, Inc., Cincinnati OH, 1999.

"The Adaptive Modeling Language. A Technical Perspective." Promotional literature, 5 pg. TechnoSoft, Inc. http://www.technosoft.com/docs/brochure99.pdf. 1999.

AML Training Manual. Cincinnati OH: TechnoSoft, Inc., 1998.

"AML Vs CAD/CAE/CAM Systems." Promotional literature, 2 pg. TechnoSoft, Inc. http://www.technosoft.com/docs/amlvscad.pdf. 1999.

Anderson, John D. *Fundamentals of Aerodynamics*. New York: Mc-Graw Hill, 1984.

Balling, R. J. and C. A. Wilkinson. "Execution of Multidisciplinary Design Optimization Approaches on Common Test Problems," *AIAA Journal, 35*: 178-186 (January 1997).

Bickel, Robert W. Cost Analyst for C-17 Systems Program Office, Wright Patterson AFB, OH. Personal Interview. 4 January 2000.

Bishop, Jonathan A., Franklin E. Eastep, Alfred G. Striz, and Vipperla B. Venkayya. "Influence of Model Complexity and Aeroelastic Constraints on Multidisciplinary Optimization of Wings," *Journal of Aircraft, 35*: 784-789 (September-October 1998).

Blair, Maxwell. "Enabling Conceptual Design in a Technology-Driven Environment," AIAA paper 98-4741, presented at

Blair, Maxwell, Steven R. LeClair, Jeffrey V. Zweber, Adel Chemaly, "Multidisciplinary Design for Uninhabited Air Vehicles," presented at the Sixth Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 18-20 June 1997 at Cambridge, MA.

Chen, Wei and Kemper Lewis. "Robust Design Approach for Achieving Flexibility in Multidisciplinary Design," *AIAA Journal, 37*: 982-989 (August 1999).

DeLoach, Scott A. Professor of Computer Science, Air Force Institute of Technology, Wright Patterson AFB, OH. Personal Interview. 27 January 2000.

Fox, Geoffrey. *Manufacturing and the National Information Infrastructure*. Syracuse, NY: Northeast Parallel Architectures Center, Syracuse University, January 2000.

Hall, Authur D. "Three-Dimensional Morphology of Systems Engineering," *IEEE Transactions on systems Science and Cypernetics, SSC-5* (2): 156-160 (April 1969).

Hildebrant, Gregory G. and Man-bing Sze. *An Estimation of USAF Aircraft Operating and Support Cost Relations*. RAND Note N-3062-ACQ. Santa Monica: The RAND Corporation, May 1990.

Hill, J. Douglas and John N. Warfield. "Unified Program Planning," *IEEE Transactions on Systems, Man, and Cybernetics, SMC-2* (5): 610-621 (Nov 1972)

Hess, R. W. and H. P. Romanoff. *Aircraft Airframe Cost Estimating Relationships: All Mission Types*. RAND Note N-2283/1 AF. Santa Monica: The RAND Corporation, December 1987.

Hess, R. W. and H. P. Romanoff. *Aircraft Airframe Cost Estimating Relationships: Bombers and Transports*. RAND Note N-2283/3 AF. Santa Monica: The RAND Corporation, December 1987.

Jameson, Anthony. "Re-Engineering the Design Process Through Computation," *Journal of Aircraft, 36*: 36-50 (January-February 1999).

Kroo, Ilan. *Computation-Based Design – A White Paper*. Stanford, CA: Aircraft Aerodynamics and Design Group, Aeronautics and Astronautics Department, Stanford University, 20 September 1996.

Martin, James N. *Systems Engineering Guidebook A Process for Developing Systems and Products*. New York: CRC Press, 1997.

Mosard, Gil R. "A Generalized Framework and Methodology for Systems Analysis," *IEEE Transactions on Engineering Management, EM-29* (3): 81-87 (August 1982).

NASA Langley Research Center. (publicly releasable portions of) *Blended Wing Body Technology Study, NAS1-20275*, CRAD-9405-TR-3780. Long Beach, CA: The Boeing Company, 1997.

Neill, D. J. and D. L. Herendeen. *ASTROS Enhancements, Volume 1 – ASTROS User's Manual*. Torrance, CA: Universal Analytics, May 1995.

Nicolai, Leland M., *Fundamentals of Aircraft Design*. California: METS, Inc. 1988.

Rais-Rohani, Masoud and George R. Hicks. "Multidisciplinary Design and Prototype Development of a Micro Air Vehicle," *Journal of Aircraft, 36*: 227-234 (January-February 1999).

Rajadas, John N., Ralph A. Jury IV, and Aditi Chattopadhyay. "Enhanced Multiobjective Technique for Multidisciplinary Design Optimization," *Journal of Aircraft, 35*: 828-830 (September-October 1998).

Raymer, Daniel P. *Aircraft Design: A Conceptual Approach*. Washington, D. C.: American Institute of Aeronautics and Astronautics, Inc., 1989.

Rechtin, Eberhardt. *Systems Architecting Creating and Building Complex Systems*. Upper Saddle River, NJ: Prentice Hall PTR, 1991.

Rowell, Lawrence F., Robert D. Braun, John R. Olds, and Resit Unal. "Multidisciplinary Conceptual Design Optimization of Space Transportation Systems," *Journal of Aircraft, 36*: 218-226 (January-February 1999).

Sage, Andrew P. "A Case for a Standard Systems Engineering Methodology," *IEEE Transactions on Systems, Man and Cybernetics, SMC-7* (7): 499-504 (July 1977)

Sage, Andrew P. *Methodology for Large Scale Systems*. New York: McGraw Hill Book Company, 1997.

Sevant, Natasha E., Malcolm I. G. Bloor, and Michael J. Wilson. "Cost-Effective Multipoint Design of a Blended High-Speed Civil Transport," *Journal of Aircraft, 36*: 642-649 (July-August 1999).

Simos, Dimitri. Lissis Home Page. Excerpt from unpublished article, n. pag. http://www.lissis.demon.co.uk. August 1998.

Smith, Howard, and others. *Blended Wing Body Study*. Cranfield, UK: Cranfield College of Aeronautics, 999.

Strutzenberg, Tyson W. "Benchmarking Multidisciplinary Center Assets". Draft Report AFRL-VA-WP-TM-1999-XXXX. Air Force Research Laboratory, Wright Research Site, Wright Patterson AFB OH, 1999.

Unix Support Advisory Group. *Report on Unix Support*. Waterloo, Ontario, Canada: University Computing Committee, University of Waterloo, 29 February 1996.

Vanderplaats, G. N. "Structural Design Optimization Status and Direction," *Journal of Aircraft, 36*: 11-20 (January-February 1999).

VITA

Lieutenant Mehmet Fidanci was born on 20 March 1971 in Kayseri. He graduated from Maltepe Military High School in 1989 and entered undergraduate studies at the Turkish Air Force Academy in Istanbul, Turkey. He graduated with a Military of Science degree in Aerospace Engineering August 1993. Upon graduation, he was commissioned a Second Lieutenant in Turkish Air Force.

His first assignment was at 2nd Main Jet Base, Izmir Turkey for Basic Jet Pilot Training. His second assignment was to Main Technical Training Base, again in Izmir Turkey. His next assignment was to Ahlatlibel Ground Radar Site, Ankara Turkey. In August 1998, he entered the School of Engineering, Air Force Institute of Technology to pursue his Master of Science degree in Systems Engineering.

Permanent Address: Keykubat Mah.
Anafartalar Cad.
67 Evler No: 38
Kayseri, 38070
TURKIYE (TURKEY)

Captain Jeffrey R. Miller was born on 13 March 1973 at Fort Sam Houston in San Antonio, Texas. He graduated from George G. Meade High School in Fort Meade, Maryland in 1991 and entered undergraduate studies at Cedarville College in Cedarville, Ohio the same year. He graduated with a Bachelor of Science degree in Mechanical Engineering in December 1995. He received his commission on 9 December 1995 through the Reserve Officer Training Corps.

His first assignment was at Los Angeles AFB, California as a project officer in the Space and Missile Systems Center's Satellite and Launch Control Systems Program Office. In August 1998, he entered the School of Engineering, Air Force Institute of Technology. His follow on assignment is to Headquarters, Air Force Operational Test and Evaluation Center (AFOTEC), Kirtland AFB, NM.

Permanent Address: 138 Natalie Circle NE
Palm Bay, Florida 32907

Captain Douglas J. Strauss was born in Toledo, Ohio in October. He graduated from Millville Senior High School in Millville, New Jersey in June 1990. In September 1990, he entered Leland Stanford Junior University in Palo Alto, California. He graduated from Stanford in June 1994 with a Bachelors of Science in Mechanical Engineering. He received his commission through Air Force Reserve Officer Training Corps Detachment 045, San Jose State University, on 11 June 1994.

Captain Strauss's first assignment was to Phillips Laboratory, Hanscom AFB, Massachusetts, where he worked in the Space Experiments Directorate as a mechanical engineer. From June 1995 to September 1996 he worked as a systems engineer for the Countermeasures Hands-On Program (CHOP) at Kirtland AFB, New Mexico, designing and testing proof-of-concept experiments for the Ballistic Missile Defense Organization (BMDO). From September 1996 to August 1998 he was assigned to the Hanscom Research Site, Air Force Research Laboratory, as a test engineer, responsible for developing instruments used for remote and in situ atmospheric characterization. In August 1998, he began his studies in Systems Engineering at the Air Force Institute of Technology. His follow on assignment is to Headquarters, Air Force Operational Test and Evaluation Center (AFOTEC), Kirtland AFB, NM.

Permanent Address: 1306 Goldfinch Ln
Millville, NJ 08332

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 2000	Master's Thesis

4. TITLE AND SUBTITLE

INTEGRATING AUTOMATED MULTI-DISCIPLINARY OPTIMIZATION IN PRELIMINARY DESIGN OF NON-TRADITIONAL AIRCRAFT

5. FUNDING NUMBERS

6. AUTHOR(S)

Mehmet Fidanci, 1LT, TUAF

Jeffrey R. Miller, Capt, USAF

Douglas J. Strauss, Capt, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology

Graduate School of Engineering and Management

2750 P Street

Wright-Patterson AFB, OH 45433

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GSE/ENY/00M-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Dr. Vipperla Venkayya, AFRL/VASD

2130 8th Street

Wright-Patterson AFB, OH 45433

(937) 255-2582

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

Advisor: Lt Col Ernest P. Smith, AFIT/ENY, (937) 255-6565 ext 4318, ernest.smith@afit.af.mil

12a. DISTRIBUTION AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Current methods of aircraft conceptual design lack the ability to quickly generate detailed analysis, particularly of nontraditional designs such as blended wing body craft. This study developed a method to resolve this problem by creating a flexible, parametrically driven conceptual model in an object-oriented, adaptive modeling environment from which analysis and optimization may rapidly be performed. These object-oriented techniques are incorporated into a traditional conceptual design process. All objects inherit dependency-tracking and demand-driven calculations.

Design Analysis was performed within the modeling language and utilized interfaces to other software packages. A detailed mesh, suitable for input into finite element analysis programs, was developed from the less detailed, geometric mesh created by the modeling program. The output from finite element analysis forms the basis for rapid changes in subsequent iterations of the design process.

The demonstration focuses on a single parametric design model which transforms a conventional transport design into a blended wing body design. This single design is controlled by a limited set of geometric variables and produces optimal structural weight estimations while the designer addresses volumetric and cost requirements.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Aircraft design, preliminary design, multi-disciplinary optimization (MDO), computer aided design (CAD), structural optimization	223
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Standard Form 298 (Rev. 2-89) (EG)
Prescribed by ANSI Std. 239.18
Designed using Perform Pro, WHS/DIOR, Oct 94